SPECIAL EDITION LIGHT & FAST DISTROS

www.linuxuser.co.uk

Linux User & Developer™

THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION

OPTIMISE LINUX

LEANER
LIGHTER
FASTER

Go light! Make Linux perform better on YOUR setup today

EXPERT FEATURE

The Future of Moore's Law

If it's dead, what happens next?

DEVELOPER GUIDE

Pick the perfect IDE

15 editors and tools to try

HEAD-TO-HEAD

VIRTUALBOX VS VMWARE WORKSTATION

DEV INTERVIEW

Nate Graham on KDE:
"We want our software to be a lean, mean productivity machine"

45 PAGES OF TUTORIALS

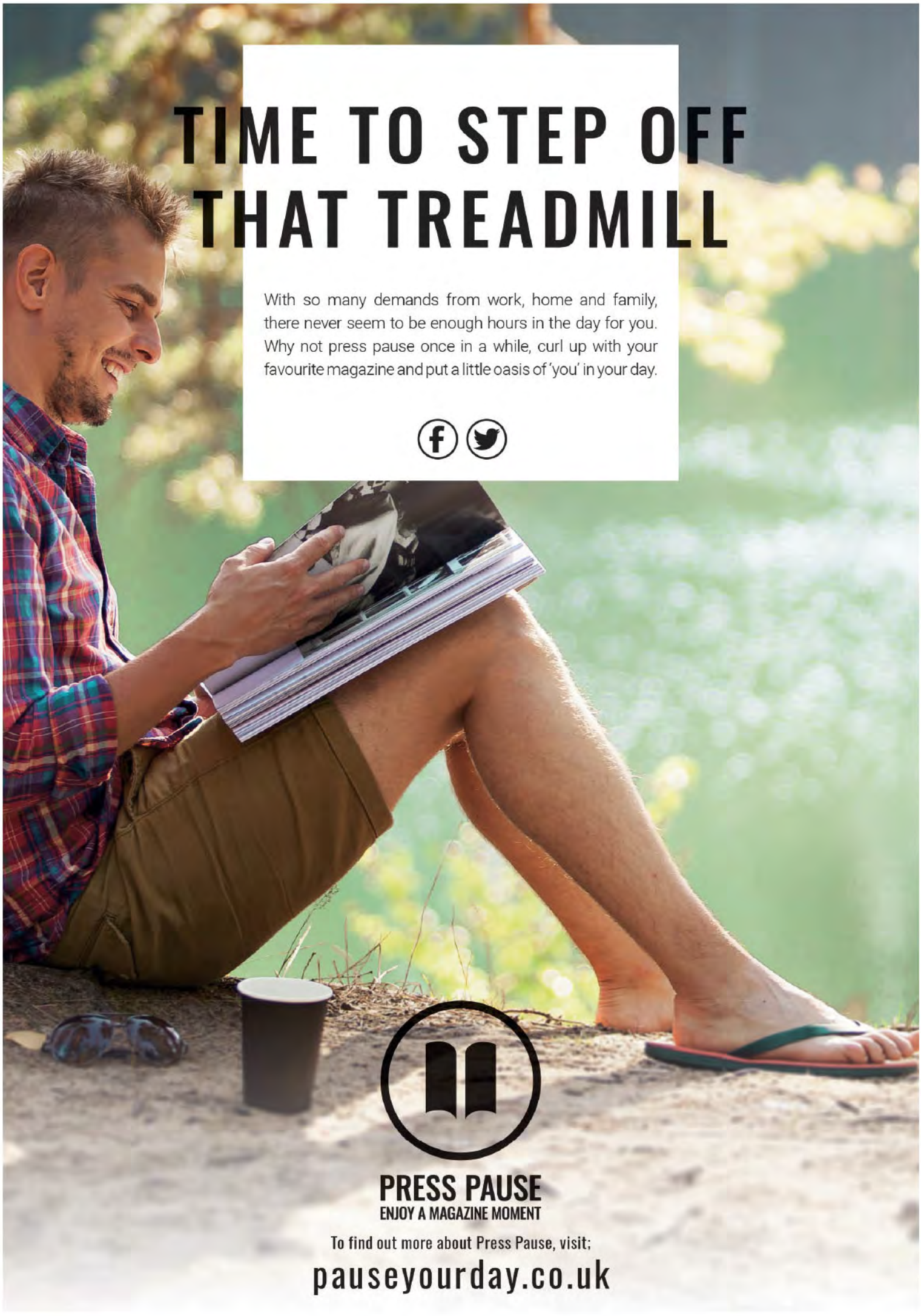
> S-TUI: Visualise your stress tests > Run .NET Core apps on the Pi > Essential Bash > Learn to program in Go > PyInstaller: share your projects

Disc cloning tools
Group Test Image and roll out multiple machines with ease

Secure your system
Expert Guide What hackers look for and what you need to change now

Kernel column
Explained L1 terminal fault + the latest kernel dev news

FUTURE


TIME TO STEP OFF THAT TREADMILL
With so many demands from work, home and family, there never seem to be enough hours in the day for you. Why not press pause once in a while, curl up with your favourite magazine and put a little oasis of 'you' in your day.
PRESS PAUSE
ENJOY A MAGAZINE MOMENT
To find out more about Press Pause, visit;
pauseyourday.co.uk

LinuxUser & Developer
THE MAGAZINE FOR THE GNU GENERATION

Future PLC Quay House, The Ambury, Bath BA1 1UA

Editorial
Editor **Chris Thornett**
chris.thornett@futurenet.com
01202 442244
Designer **Paul Tysall**
Production Editor **Ed Ricketts**
Editor in Chief, Tech **Graham Barlow**
Senior Art Editor **Jo Gulliver**

Contributors
Dan Aldred, Mike Bedford, Joey Bernard, Chris Binnie, Christian Cawley, Neil Bothwick, John Gowers, Toni Castillo Girona, Tam Hanna, Georgine Hodsdon, Lewis Leong, Jon Masters, Paul O'Brien, Calvin Robinson, Mayank Sharma, Shashank Sharma, Kevin Wittmer

All copyrights and trademarks are recognised and respected. Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

Advertising
Media packs are available on request
Commercial Director **Clare Dove**
clare.dove@futurenet.com
Advertising Director **Richard Hemmings**
richard.hemmings@futurenet.com
☎01225 687615
Account Director **Andrew Tilbury**
andrew.tilbury@futurenet.com
☎01225 687144
Account Director **Crispin Moller**
crispin.moller@futurenet.com
☎01225 687335

International
Linux User & Developer is available for licensing. Contact the International department to discuss partnership opportunities
International Licensing Director **Matt Ellis**
matt.ellis@futurenet.com

Subscriptions
Email enquiries **contact@myfavouritemagazines.co.uk**
UK orderline & enquiries **0344 848 2852**
Overseas order line and enquiries **+44 (0)344 848 2852**
Online orders & enquiries **www.myfavouritemagazines.co.uk**
Head of subscriptions **Sharon Todd**

Circulation
Head of Newstrade **Tim Mathers**

Production
Head of Production US & UK **Mark Constance**
Production Project Manager **Clare Scott**
Advertising Production Manager **Joanne Crosby**
Digital Editions Controller **Jason Hudson**
Production Manager **Nola Cokely**

Management
Managing Director **Aaron Asadi**
Editorial Director **Paul Newman**
Head of Art & Design **Rodney Dive**
Commercial Finance Director **Dan Jotcham**

Printed by
Wyndeham Peterborough, Storey's Bar Road, Peterborough, Cambridgeshire, PE1 5YS

Distributed by
Marketforce, 5 Churchill Place, Canary Wharf, London, E14 5HU
www.marketforce.co.uk Tel: 0203 787 9001
ISSN 2041-3270

We are committed to only using magazine paper which is derived from responsibly managed, certified forestry and chlorine-free manufacture. The paper in this magazine was sourced and produced from sustainable managed forests, conforming to strict environmental and socioeconomic standards. The manufacturing paper mill holds full FSC (Forest Stewardship Council) certification and accreditation

All contents © 2018 Future Publishing Limited or published under licence. All rights reserved. No part of this magazine may be used, stored, transmitted or reproduced in any way without the prior written permission of the publisher. Future Publishing Limited (company number 2008885) is registered in England and Wales. Registered office: Quay House, The Ambury, Bath BA1 1UA. All information contained in this publication is for information only and is, as far as we are aware, correct at the time of going to press. Future cannot accept any responsibility for errors or inaccuracies in such information. You are advised to contact manufacturers and retailers directly with regard to the price of products/services referred to in this publication. Apps and websites mentioned in this publication are not under our control. We are not responsible for their contents or any other changes or updates to them. This magazine is fully independent and not affiliated in any way with the companies mentioned herein.

If you submit material to us, you warrant that you own the material and/or have the necessary rights/permissions to supply the material and you automatically grant Future and its licensees a licence to publish your submission in whole or in part in any/all issues and/or editions of publications, in any format published worldwide and on associated websites, social media channels and associated products. Any material you submit is sent at your own risk and, although every care is taken, neither Future nor its employees, agents, subcontractors or licensees shall be liable for loss or damage. We assume all unsolicited material is for publication unless otherwise stated, and reserve the right to edit, amend, adapt all submissions.

FUTURE
Connectors. Creators. Experience Makers.

Future plc is a public company quoted on the London Stock Exchange (symbol: FUTR)
www.futureplc.com

Chief executive **Zillah Byng-Thorne**
Non-executive chairman **Peter Allen**
Chief financial officer **Penny Ladkin-Brand**
Tel +44 (0)1225 442 244

Welcome

to ~~issue 190~~ of Linux User & Developer

the last issue

In this issue

» Optimise Linux, p18
» Pick the Perfect IDE, p56
» The Future of Moore's Law, p64

Sad news, readers. After almost 19 years, this will be last issue of Linux User & Developer. While that sinks in, the team would like to take the opportunity to thank you all for reading and being part of our adventures in Linux and the world of open source and free software.

The magazine began in 1999 and was handed out to attendees of Linux Expo, and moved to the newsstand a few years later. We recently spoke to Maggie Meer, the original publisher of **LU&D**, and she recalled how passionate people were and how producing the title felt like being part of a vast community. That sense of community is something we take for granted now, so as **LU&D**'s little part in Linux history ends, we want to encourage you to get involved in helping the open source software you care about. If you're a fan of a distro or app, consider bug testing early releases or donating to their development; if you think something is missing from the scene – and it's probably not a music player – then try coding it. You are what makes Linux what it is, and together you can change the world! Happy venturing. It's been a pleasure!
Chris Thornett, Editor

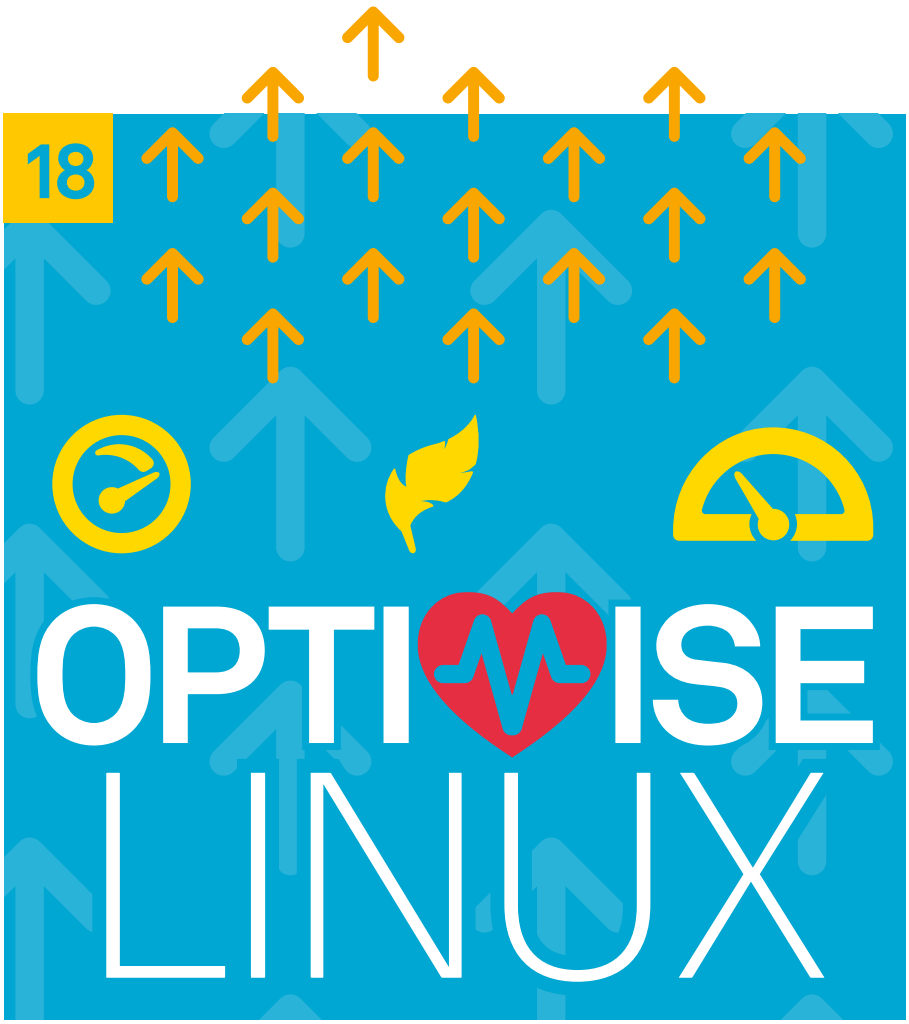
Get in touch with the team:

linuxuser@futurenet.com

Twitter: @linuxusermag
Facebook: facebook.com/LinuxUserUK
Filesilo: filesilohelp@futurenet.com

SUBSCRIPTION OFFER
Subscribe and receive a FREE Native Union Eclipse charger hub!
FREE GIFT!
MORE DETAILS ON PAGE 30

Contents



OpenSource

06 News
Redis abandons open source in its database modules. Plus: Steam!

10 Letters
Your final batch of beautiful words

12 Interview
KDE developer, Nate Graham on the project's bug-fixing blitz

16 Kernel Column
Jon Masters on the latest updates

SpecialReport

32 Coding a murderer
How the Murder Accountability Project is helping to catch killers

Features

18 Optimise Linux
Mainstream Linux distros have to appeal and work with a wide range of users and hardware, so they're not always the fastest. Our guide shows you how to streamline and optimise a distro for your own use case

56 Pick the Perfect IDE
Linux provides a powerful platform for software developers and offers a wide choice of Integrated Development Environments. Discover which one best meets your needs

64 The Future of Moore's Law
Is Gordon Moore's famous dictum really no longer applicable – and if so, what can we expect from the hardware of the future?

Tutorials

36 Essential Linux: cfdisk
Learn how to use cfdisk to partition disks, and how to install Linux

40 S-TUI: Stress-test your CPU
Use S-TUI to monitor your CPU and run a stress test from the command line

42 CompSec: Plug those gaps
Fix the more obvious threats to your system and that includes printers

46 Python: Pyinstaller
How to more easily share your Python projects using various utilities

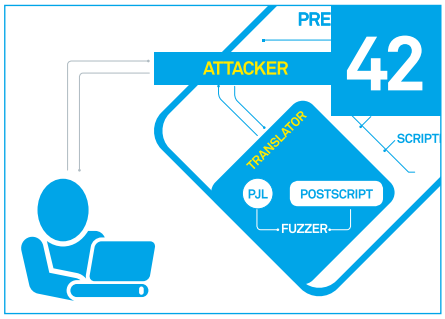
50 Programming: Go
In part 3, we integrate an SQL database with our video-sharing server to enable keyword searching

LinuxUser & Developer
Issue 196
September 2018
facebook.com/LinuxUserUK
Twitter: @linuxusermag

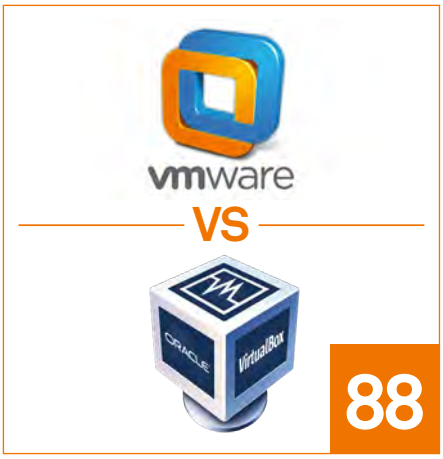
FileSilo
94 Free downloads
We've uploaded a host of new free and open source software this month

70

86



Raspberry Pi .NET Core
Run .NET Core web apps on the Raspberry Pi
Microsoft officially extends Linux support to the Pi – find out how to use it
Kevin Wittmer
.NET CORE
Resources
76



88

Practical Pi

70 Introducing MidiIdentifier
By hooking up a MIDI keyboard to a Pi, **Andreas Scholz** and **Karsten Schick** created a piano that attempts to guess what you're playing...

72 Build an image auto-uploader for the Raspberry Pi
Your phone can automatically upload pictures to the cloud, so why not the Pi Camera? Learn how to build and code a Dropbox-enabled solution in Python

76 Run .NET Core web apps on the trusty Pi
With Microsoft's .NET Core now supporting ARM chips, it's possible to run it on a reasonably recent Pi. Discover how to get everything set up for coding .NET apps

Reviews

81 Group test: Disk cloning tools
How well do Clonezilla, FOG Project, G4L and Redo Backup & Recovery perform when it comes to imaging and rolling out multiple machines at once?

86 Hardware
Beyerdynamic's new Amiron Wireless headphones promise audio perfection over Bluetooth – but you may need a second mortgage to buy them

88 VMware Workstation 14.1.3 vs VirtualBox 5.2.18
One is open source, the other is proprietary, but which one is better?

90 Fresh FOSS
HSTR 1.27.0, Mattermost 5.2, Tardis 1.0.2 and Audacious 3.10 reviewed

Back page

96 Goodbye, dear reader
A montage of some of the many covers we've produced over the years.

SUBSCRIPTION OFFER
SUBSCRIBE TODAY
Get a free high-speed USB charger when you subscribe!
Turn to page 30 for more information
FREE GIFT!

BUSINESS

Database maker Redis makes its module code proprietary

A timely reminder that open source is not for life...

Redis Labs has announced that its range of Redis Modules are no longer open source. By abandoning the AGPL licence, in favour of combining Apache v2.0 with Commons Clause, in-house modules extending the functionality of the Redis database software are now off-limits to third-party cloud service providers.

Modules affected by this change include RediSearch, Redis Graph and ReJSON. Redis Labs cofounder and CTO Yiftach Shoolman justified the development in a blog post, complaining that "Cloud providers have been taking advantage of the open source community for years [...]. This discourages the community from investing in developing open source code, because any potential benefit goes to cloud providers rather than the code developer."

To understand the implications of this, we spoke to Shashank Sharma (**www.menschlaw.in**), a trial lawyer and software licensing expert who has been writing about open source software for more than 14 years.

He told us that while the Redis project remains BSD-licensed, only RedisLabs' Redis Modules have been switched to the "so-called Apache+Commons Clause license. [However,] the Redis community can still contribute modules to the project, under their own chosen licences, such as AGPL. In fact, the lead developer in a blog post confirmed that his current Redis module, Disque, will be released under AGPL."

The important distinction, notes Sharma, is that "just because the source is available doesn't make software open source or free. There are essential freedoms (as the FSF calls them) which makes software Free."

> "Just because the source is available doesn't make software open source or free"

Similarly, the Open Source Definition has a set of conditions defining open source software. By adopting the Commons Clause, Redis is eroding, or fundamentally altering, these freedoms. "Redis call the new licence for Redis Modules Apache + Commons Clause, which may lead to people mistakenly concluding that Redis Modules are still open source. Any alteration of the licence terms to limit the freedoms/permissions that it allows changes the licence from open source to proprietary."

Redis claims it wants to limit how cloud providers can profit from Redis deployments and support services, ensuring any money made with its software isn't by third parties. But Sharma told **Linux User & Developer** that the adoption of the Commons Clause has "broader implications," referring specifically to this line of the clause:

"'Sell' means practising any or all of the rights granted to you under the Licence to provide to third parties, for a fee or other consideration (including without limitation

"Commons Clause" License Condition v1.0

The Software is provided to you by the Licensor under the License, as defined below, subject to the following condition.

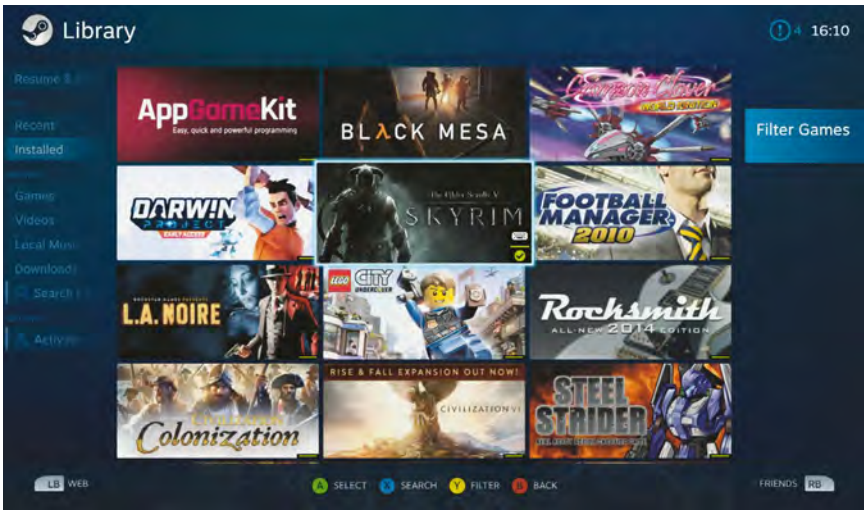
Without limiting other conditions in the License, the grant of rights under the License will not include, and the License does not grant to you, the right to Sell the Software.

For purposes of the foregoing, "Sell" means practicing any

Above Redis says it doesn't want third-party cloud providers profiting from its software

fees for hosting or consulting/support services related to the Software), a product or service whose value derives, entirely or substantially, from the functionality of the Software."

Not only will this impact cloud providers, but also Linux support companies and individual consultants. Sharma concludes: "The real issue for anyone using Redis is how much of the reliance is on Redis Core, and whether anyone can use Redis without relying on Redis Modules."



GAMING

Steam Play brings more Windows games to Linux

Modified Wine and new DirectX layer promises great Linux gaming

Just months after it seemed Valve might have given up on Linux gaming, a new beta version of Steam Play has been released, bringing compatibility to a big selection of previously Windows-only games. As well as giving Linux Steam users access to a larger catalogue of games, Valve hopes that developers will be able to more easily port projects from other platforms.

Whether you use the Steam client on your Linux PC or you're running Valve's own Linux-based SteamOS, this development means that a number of top games, such as Star Wars: Battlefront 2, Tekken 7, Fallout Shelter and most of the Doom titles (including the 2016 VR version) can now run on Linux.

Steam Play supports Windows games which don't have a native Linux version to be installed and run within the Linux Steam client. This is made possible thanks to a two-tier approach.

First is the work Valve has done with Codeweavers in developing Proton, a modified version of the Wine compatibility layer. Valve's announcement post explains that "modifications to Wine are submitted upstream if they're compatible with the goals and requirements of the larger Wine project; as a result, Wine users have been benefitting from parts of this work for over a year now."

Second is compatibility with Direct3D graphics, thanks to vkd3d. This implementation of Direct3D 12 uses Vulkan for high-performance graphics. DXVK is also included – another Vulkan implementation, this time for Direct3D 11. This has involved "direct support from our open-source graphics group to fix Mesa driver issues affecting DXVK," and collaboration with "Khronos, NVIDIA, Intel and AMD to coordinate Vulkan feature and driver support."

Meanwhile, to improve the overall gaming experience, full-screen graphics and multithreading have all been enhanced. Full-screen games will now be "seamlessly stretched to the desired display without interfering with the native monitor resolution or requiring the use of a virtual desktop." Additionally, all Steam-supported controllers should now be automatically recognised by supported games.

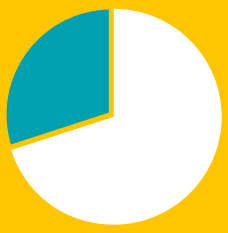
DISTRO FEED

Top 10

(Average hits per day, 30 days to 24 August)

1.	Manjaro	▲	3691
2.	Mint	▼	2080
3.	MX Linux	▲	1884
4.	elementary	▼	1818
5.	Ubuntu	▲	1447
6.	Debian	◀▶	1028
7.	Zorin	▲	869
8.	Fedora	▲	718
9.	openSUSE	▲	667
10.	Solus	▲	631

This month



In development (3)
Stable releases (7)

The Arch Linux-based Manjaro continues to dominate the Linux scene, but distros from the Debian family continue to dominate Linux overall.

Highlights

ArcoLinux
Previously known as ArchMerge, this distro is available in a choice of two projects: the main desktop option and a minimal option.

SwagArch
This German-based distro runs Xfce and uses the Calamares graphical system installer. A number of open source apps are preinstalled.

ArchMan/GNU Linux
Again using Calamares, this Turkish Linux distro includes a preconfigured desktop environment. The Octopi package manager is also included.

Latest distros available:
filesilo.co.uk

HARDWARE

Lenovo finally joins the Linux Vendor Firmware Service

ThinkPad firmware updates will soon be easier to install

Running a Linux-powered ThinkPad, the laptop range launched by IBM and acquired by Lenovo in 2005, is a generally pleasing experience. However, without the right firmware updates, which need to be sought out, it's unlikely to be perfect. Thanks to Lenovo joining the Linux Vendor Firmware Service, this has just become a lot easier.

The LVFS enables equipment manufacturers to distribute UEFI firmware directly to users, while making the process as safe and reliable as possible, with the minimum of reboots. According to the LVFS introduction at **https://fwupd.org**, "The LVFS runs on a dedicated virtual private server and uses a public CDN for metadata. Every month there are over 50 million files being downloaded from 10 million clients. There are currently 281 devices supported on the LVFS with 758 available firmware versions."

Various manufacturers are supporting Linux in this way, but not all. Announcing the news in his blog, Red Hat's Richard Hughes noted: "I do think Lenovo should be applauded for the work done so far due to the enormity of the task, rather than chastised about coming to the party a little late. If anyone from HP is reading this, you're now officially late."

Above Lenovo's popular Linux ThinkPad line will benefit from LVFS membership

Bringing Lenovo into the LVFS hasn't just happened overnight. Many changes have been made to accommodate the move, such as "...changes to the low level fwupdate library, fwupd, and even the LVFS admin portal itself for various vendor-defined reasons." The LVFS only supports Lenovo hardware "produced fairly recently" however; "There's no point looking for updates on your beloved T61" is the official advice. There's also little word on what devices from the Chinese manufacturer will be supported in future, but it's certainly a step in the right direction for now.

DISTRO

Chrome OS 69 to include Linux support

Six months on from the announcement of Project Crostini, Chrome OS 69 is set to introduce the 'Linux (Beta) for Chromebooks' feature in its 4 September release. Employing virtual machines and sandboxed containers, the feature enables the installation of Linux apps, development environments and other utilities on Google's Chrome OS.

As the release notes for Chrome OS 69 explain: "Linux (Beta) for Chromebooks allows developers to use editors and command-line tools by adding support for Linux on a Chrome device. After developers complete the set up, they'll see a terminal in the Chrome launcher. Developers can use the terminal to install apps or packages, and the apps will be securely sandboxed inside a virtual machine."

Once running, Chrome OS users will be able to install these apps from the launcher, just like any other. While basic apps should work without a problem, those requiring accelerated graphics and video decoding are unlikely to deliver at this stage.

HD video editing applications and graphics-intensive video games from Steam are thus unlikely to work at this stage, regardless of hardware. Older games and those without graphics acceleration should run, however.

Software that can be expected to run based on success under Crostini includes the image manipulation toolw GIMP and Inkscape, as well as the Kdenlive video editing suite. Meanwhile, LibreOffice should also run, providing a strong alternative to the limits of Google's cloud-based office suite.

A list of supported devices – that is, those that run virtual machines – can be found at **www.chromium.org**, with Chromebooks from Acer, HP, and Lenovo among them.

OPINION

Security automata

What can you do when infrastructures are too performant for humans to monitor effectively?

The cloud is all the rage and it has been, in one form or another, for at least a decade. Vendors and organisations alike continue to get caught up in what we now call cloud-native technologies and their many facets. The CNCF (Cloud Native Computing Foundation) helps to propagate the adoption of cloud-native tech and "open source, vendor-neutral projects" in order to improve the accessibility of innovations in the cloud space.

According to the CNCF, the definition of cloud-native relates to scalable architecture which is both resilient and which offers the ability to make changes to critical, live services seamlessly. As such, "containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach."

However with innovation come new attack vectors, which need to be carefully considered by all parties involved. The result is that old-school security thinking no longer applies. The approach of "We'll just wade through the seemingly endless code in our (micro)services and then secure and remediate any issues" simply doesn't wash in the new cloud-native world we live in.

Numerous security tools have sprung up from varying vendors to help deal with the new and revised attack surfaces which now pose a security risk to organisations. But while auditing in an automated manner to check for changes to your cloud code is a valuable exercise, there's still a degree of operational monitoring and human interaction required.

In the high-paced, dynamic world of cloud apps, just auditing your security issues sadly doesn't get anywhere near to cutting the mustard. Thanks to the many events taking place during any given minute – such as containers automatically spinning up as soon as another fails, subtle automated changes to network security groups and encrypted secrets being passed back and forth – to my mind we've already created infrastructure that is too performant for humans to monitor in an effective manner.

The solution is actually relatively straightforward; rolling it out across a critical server estate is however much easier said than done. Two tools which can provide the required functionality are Cloud Custodian (**https://capitalone.github.io/cloud-custodian/docs**) and Dome9 (**https://dome9.com**). Among their feature-sets is the ability to highlight and then automatically remediate security and compliance issues based on a set of predetermined rules.

Cloud Custodian, for example, has the ability to power down EC2 instances for patching runs, shout loudly about logins by the root user, and take care of storage volumes which aren't attached to anything useful any longer. It can additionally alert and then delete gaping-wide open security groups (those boasting 0.0.0.0/0 IP Address ranges, for example) and also notify users when their access keys need rotating – and deny the user access entirely if they don't comply within a reasonable time!

In the same vein, the CloudBots framework from Dome9 also provides "active remediation". In a busy working environment where new software features keep the lights on, and therefore the bosses happy, it's common for security and housekeeping (which sometimes introduces other security implications) to slip down priority lists. Thankfully both tools emphasise their ability to automate away these concerns, which means that key issues are dealt with promptly and without much fanfare in most cases.

> "With innovation come new attack vectors, which need to be carefully considered by all parties"

Clearly a high degree of testing is required before actions are written into rules which affect production environments, but in the case of development and staging environments, the testing is more accessible. The fact that these tools also run off coded rulesets means of course that the GitHubs and BitBuckets of this world can be successfully used for revision control and sanity testing.

Let's not forget, either, that misconfiguration is also the bane of a security professional's life, and thus some automated sweeping-up here and there is an undeniably welcome extra functionality.

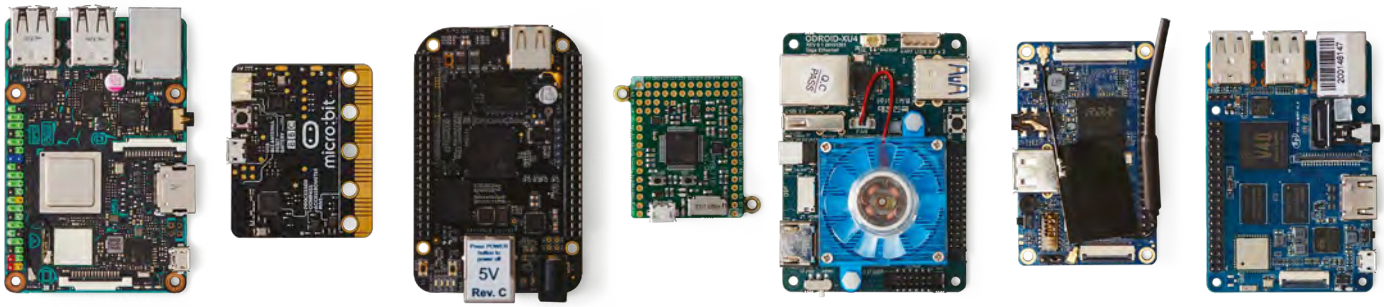
In short, you could say that we're already at the point where we need computers to run our other computers. We may be 21 years down the line from when Skynet was first meant to be activated, but let's not forget that the damned thing learns fast...

Chris Binnie
Chris' book *Linux Server Security: Hack and Defend* teaches you to make servers invisible and mitigate attacks. See **www.devsecops.cc**

COMMENT

Your letters

Questions and opinions about the mag, Linux and open source



Above Linux and small board computers go together well, and tinkering with them both can open up the prospect of a career

TOP TWEET

Not a bad tweet to end on… Edward Snowden reminds us how important Linux is for users seeking a secure operating system.

"@Snowden If you're serious about security, @QubesOS is the best OS available today. It's what I use, and free. Nobody does VM isolation better."

Linux 4 life

Dear LU&D, I was introduced to Linux many years ago for a game project, but just did the work on a small system and didn't pay much attention to it. A few years ago I started doing contract work for Google ATAP (Advanced Technology and Projects) that involved prototyping on the Pi, which started a whole new chapter in my life.

I've been working with them ever since and it's my favourite job, but the Linux side has taken on a life of its own. I dove in head first to learn everything I could about IoT devices/marketplace and Linux has been a major part of it. I've been working my way through the popular IoT electronics (MCUs, SBCs, displays, sensors and motor controllers and so on) by buying the part, reading the datasheet, writing custom code and then publishing it on GitHub (**https://github.com/bitbank2**).

I've learned a ton and have come to appreciate the power and simplicity of Linux. I'm able to write portable C code that can run on any CPU from a $7 ARM board to a $3,000 laptop. Now I've started to use my Linux experience and IoT knowledge to expand my client list and get new jobs in both software and hardware development.

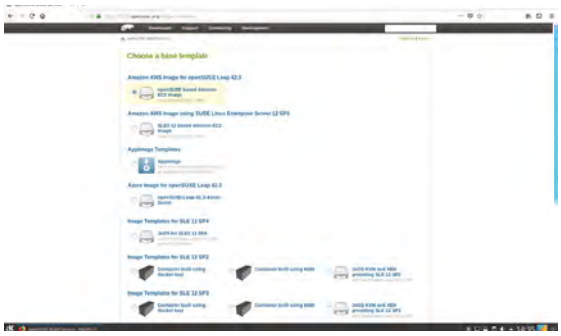
Laurence Bank

Chris: Thanks, Laurence. Everyone's journey with Linux is different and it's exciting to think that 25 years on from the release of the Linux kernel, it's commonplace to see people making a living from Linux in all its many, many forms. If you've not come across Google ATAP (**https://atap.google.com**), it's worth a look. Being interested in next-gen interfaces as I am, Project Soli is particularly fascinating as it demonstrates an interaction sensor chip using radar technology. The chip is able to pick up tiny finger movements, such as simulating the turn of a dial with a finger and thumb.

Fred Linux

Dear LU&D, I don't know if anyone has mentioned it already, but I would love to learn how to make my own GNU/Linux distribution and maybe desktop environment in a future issue. Not necessarily writing one from scratch, but creating bootable media of your current system and making it installable.

Frederik Scholl



Above You can still use Open Build Services while it and SUSE Studio are merging, but Bodhi Builder is a good option for Ubuntu

Right Chromebook x2s are as rare as hen's teeth (excluding the Talpid mutant chicken, of course), although we have been told that John Lewis has the exclusive in the UK

Chris: We get this request quite a lot, Frederik. Quite a number of the tools for building your own Linux distro have disappeared over time. (We also notice that OpenSUSE's Studio and Open Build Services are merging to become OpenSUSE Express and creating quite a major upheaval in the process.) Probably the most popular tool (or script) was remastersys, and although that has ceased development many years ago, a number of distro development teams have taken on the script and reworked it into their own tools. We actually cover how to take a base Ubuntu, get the packages you want and create a custom ISO with Bodhi Builder (see p24). Hopefully, that will get started down the road to getting up the custom distro you want!

X marks the X2

Dear LU&D, I read your review of the HP Chromebook x2. Can you please advise where one can be obtained in the UK? Your review says **http://www.acer.co.uk**, which surely must be a typo?
Yannis Kingdom (via Twitter)

Chris: Yes, we imagine trying to find an HP Chromebook on the Acer site would be quite fruitless... Sorry for that blooper. That comes from using an old template and no one noticing throughout the entire production process. The review of the x2 was quite an opportunity for us as UK availability has been very low, but the few reviews that we'd read were glowing. We managed to get one from HP US, but before we ran the review we received confirmation that the Chromebook x2 would be available in the UK as a John Lewis exclusive. However, this doesn't appear to have happened yet as there's no mention of this particular Chromebook on the John Lewis website.

PARTING WORDS

It's goodbye from us...

If you didn't read the welcome page (p3) this will be our last issue. The publisher's decision to close Linux User & Developer was taken quickly, so we didn't gain an opportunity for readers to say goodbye properly, but we were able to gather a few farewell words from some regular writers:

"Writing for **LU&D** has been an exciting experience from the very start, making me proud of being a small part of something much bigger – a great community of Linux enthusiasts growing together and speaking the same language: GNU/Linux! It's been a privilege to share my humble knowledge of computer security with you all through the pages of **LU&D**. So long."
Toni Girona Castillo

"Incredibly saddened by the closure of **Linux User & Developer** after 19 years. As much as I've loved contributing features, I've enjoyed being a reader even more. It will be missed, I am sure."
Paul O'Brien

Left A few of the writers wanted to say goodbye, and we were able to find something suitable to mark its passing

INTERVIEW KDE

Project: Lean and mean

KDE is a vast project that celebrates 22 years in October. We spoke to **Nate Graham** about KDE's ongoing bug-fixing charm offensive

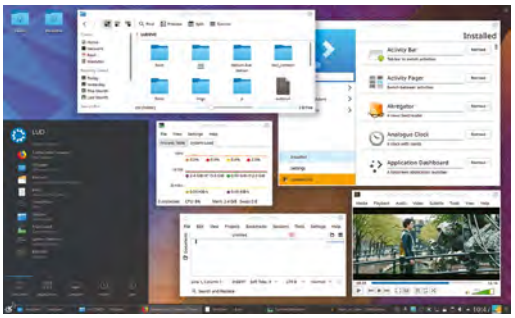
Nate Graham
Nate, a former Apple engineer, is a KDE developer and product manager for Kubuntu.

KDE developer Nate Graham spearheads KDE's usability and productivity initiative, which is one of the three major goals of the organisation that were agreed in 2017. "The thrust of this goal is we want to make KDE Plasma and our KDE apps easier to use, and more approachable for typical users," Graham told us, chatting over VoIP from his sofa at home – and the response, particularly for recent Plasma point releases, has been positive. Going further, KDE wants to make complex workflows a lot simpler and easier for anyone to use: "We want our software to be a lean, mean productivity machine," says Graham. Read our, (lightly edited) interview below:

Let's start with what you do on the KDE project...
I write documentation to help people get involved in the project. I do bug triaging, so that we can have bug-tracking systems that are accurate. I mentor for new contributors to help them get involved in this; I submit lots of patches myself. I review patches that other people have submitted. I do user interface design, and usability and visual stuff. I'm involved in this process quite a lot, and my work takes me into contact with lots of other members of the KDE community, which is one of the most rewarding parts of it, because I get to learn from them, and I hope that they get to learn from me, too.

I wondered how you ended up working on KDE from working at Apple?
It's a funny journey. Working at Apple was always one of my childhood dreams. I always loved Apple's software and hardware, because I felt that they really focused really aggressively on both the overall user experience and also the little polished details that were often overlooked in other software. They really said, "It's not okay that we get 95 per cent of the way there. We need to get 100 per cent of the way there." I worked at Apple for eight years, but eventually, it was time to move on. Once that was done, I decided that I wanted to be involved in a part of the Linux community.

So I looked around for a little while, and I tried to find a Linux project that I felt matched the most what I saw Apple was doing. And I found a number of different ones. But ultimately, the reason why I settled on KDE was not so much I felt they were currently the most Apple-like, but because I felt like KDE's community was the one that was most welcoming towards becoming this way – and the most welcoming towards me, as well.



Above Kubuntu's 18.04 new dark theme for Plasma, with a light theme for KDE apps, turned out to very popular

You put in your usability and productivity proposal last year. You've covered it a little bit, but could you tell us a little more about the initiative?
When I initially proposed this and it was accepted, I set a timeline for two years. I said to myself, 'I think we need to do basically 10 years' worth of improvement every year.' So 20 years' worth of product usability improvements in two years, maybe 30 in three years. But I think we can absolutely get there and as the excitement for this initiative has built up, I think we see an increasing number of people wanting to become involved in it, which is really rewarding – because lots of new contributors means lots of new help, and the pace of the change happens faster. Which makes everybody else excited, and it becomes a virtuous circle.

Has that led to changes in quality testing? Has it changed how you test the apps and framework?
The way we do our testing is somewhere I feel like we still have some improvements to make. This is sort of an endemic problem everywhere, because testing is one of the least fun aspects of software development, of course. And all of our software has very aggressive release schedules – we release three new versions of Plasma every year. We release three new versions of *all* our apps every year. And

QUICK GUIDE

Kubuntu and KDE: What's the difference?

There are a great number of Linux distributions that use parts of the KDE project, but the most popular distros are KDE Neon, which is listed as an official KDE product, and Kubuntu, an official Ubuntu flavour that uses the KDE Plasma desktop environment. However, there's some general confusion about how the two differ, so we asked Graham what the key differences were. "This is a question that comes up very frequently," he says. "And I think this is kind of our fault for not adequately differentiating these, or really discussing the ownership there."

As Kubuntu product manager, Graham always encourages people who want a general-purpose KDE

Above Nate Graham: "I think it's important that KDE Neon is seen as a showcase of KDE software for enthusiasts, rather than as a general-purpose OS."

desktop distribution to use Kubuntu: "Neon's stated goal and reason for existing is to provide, essentially, a rolling-release set of KDE software on top of a non-rolling-release base," explains Graham. "KDE Neon doesn't have as much of a focus on integration as Kubuntu has. So basically, if you want the latest KDE software, and you don't care the latest 'anything else', use KDE Neon.

"If you want the latest updates for everything on your distribution then a pure rolling-release distro such as Arch or Tumbleweed is the best way to go. If you want relatively stable, well-tested versions of everything, including KDE software, and you want for there to be good integration and defaults, then you get Kubuntu," states Graham.

Graham's everyday distro is actually KDE Neon: "I find that the Developer Unstable edition is a great developer platform, because it's super-rolling [and] built from Git branches every couple of days, so you've always got the latest and greatest."

we release a new version of the frameworks that support all of them once a month – so 12 times a year. Often, this doesn't leave enough testing time.

For frameworks, there's always the possibility that if something breaks or if there's a regression, we can just ship a fix the next month. But for all the other ones, it means that we have very abbreviated release schedules, which means that the demand for testers and testing is always quite extreme. One of the things we're really looking to do is to help people test the beta versions of our software.

We already have quite a number of testers who do this. Those people are amazing. I want to give a shout out to them because they are so important and awesome. But we can always use more help. Testing the beta versions of KDE software is probably one of the biggest ways that a non-software developer contributor can help to keep the quality levels up.

You've had quite a focus on Discover recently?
Yes, I feel Discover has made huge strides in the last few releases. When I started with Plasma, the feedback that I always saw with Discover, at least online from users, was quite negative – and often quite hostile, which was a little bit demoralising.

That was one of the reasons why I felt we really needed to focus on this project. It wasn't that it was a bad project: there were a few little niggling issues of the sort that tend to annoy users. Even if the foundation is really stable, if the interface isn't polished to the extent that people want or it's not totally reliable, people often find they're frustrated.

So I worked with the lead developer, Aleix [Pol], who is an amazing guy, and has been involved with KDE for a very long time. He, himself, had noticed most of these issues and was already moving towards resolving them. So I helped out, mostly on the user interface side, to make Discover look and feel more like a Desktop app when it's running on the Desktop, and to solve some user interface inconsistencies, and to propose some better behaviours, and to redesign some of the pages to look more visually attractive.

I think we saw a pretty good response for the 5.12 release. The 5.13 release, we got a really good response. And the upcoming 5.14 release, I think people are going to like it even more, because we've made even more of the kinds of changes that they want. And now Discover is stable. It has lots of features. It's very visually attractive. So personally, I think Discover is an example of a current success story, where we focused a lot of our resources aggressively on one app that needed some attention. Now that we're there, we can redirect those resources elsewhere, and move onto the next thing.

It's a very large project, so how do you deal with the interoperability and compatibility issues with KDE in terms of dependencies and consistency?
KDE Plasma and KDE apps are run on so many different distros, and some people even run some of them on Windows and Mac operating systems. We often do have lots and lots of bug reports from users. To a certain extent, this happens in lots of computer software, because developers tend to use certain

> "I want to give a shout out to our testers because they are so important and awesome"

Above KDE has recently released a Qt-based client for GNU Ring (**https://ring.cx**), the free and universal communication platform. Ring uses distributed hash table technology to create its own network. Since July 2018, GNU Ring now supports features such as audio and video call recording and push notifications

types of systems, and users tend to use a vastly more diverse set of hardware and software than we do. This is why having beta testers is so important.

I want to really thank the users of rolling release distros at this point, because to a certain extent, in the whole Linux ecosystem, they provide a release QA for everybody else. This is, I think, especially true for KDE software, because of our aggressive release schedules. Every single month, we release a new version of KDE frameworks. So that gets into the hands of Arch users practically immediately. So if we get a whole bunch of bugs from Arch users that say, 'Hey, this broke', we can fix it in the next release of KDE frameworks. And that's going to be completely invisible to the people who use Kubuntu, who use Debian, who use openSUSE. They basically benefitted from the Arch people's QA. So I'd just like to say thank you, Arch and Manjaro and Tumbleweed users. You're all doing the community, and the wider Linux ecosystem, a really incredible service.

What do you feel are the barriers for KDE?

I gave a presentation at Akademy [KDE's community gathering], and I targeted a number of user personas whom I think can benefit from KDE Plasma. For the most part, these are technical professionals, business professionals, students and teachers, people along those lines.

I think what we need to do is look at the reasons why they don't currently use our hardware and software. A major one is a lack of pre-installation on people's hardware. I think that really is the biggest one. Because most people don't buy software; they buy hardware. To most people, the physical thing they can hold in their hands is the product. So we need to get KDE Plasma and KDE apps on more hardware. But in order to do that, what we need to do is first focus on our software to the point where hardware vendors can look at it and say, 'Hey, wow, this thing is so good. I really want to put this on my hardware.' And then it'll be attractive to users. So I work backwards from there. I say, 'We've got to put it on hardware. To do that, we have to focus on software. But which ways do we focus on software?'

We need to make sure that it's meeting business and enthusiast power-user use cases, and that it's meeting student and teacher use cases. Those are the primary ones.

From that very high level perspective, that leads to some technical things. The people who I'm talking about very frequently use remote files on Samba shares, for example. This is something that technically we're a little bit behind on. Our tech for accessing files and folders on Samba shares is not a reliable as it needs to be. So that's something I would really like to focus on. I feel like if we managed to do that, we reduce the friction a little bit for the people in those use cases. Then we make the software more attractive, and the hardware vendors are going to want it, and then people are going to be able to buy stuff with it on. And then we take over the world...

As product manager for Kubuntu, how's 18.10 coming along?

Pretty well. We're going to have Qt 5.11. We're going to have Plasma 5.13. We've gotten a very good reception from Kubuntu 18.04, users are pretty happy with it. We've made a couple of very bold user interface and defaults decisions, too, that I was hoping people would respond well to – and so far, they have. This included one that I wasn't fully on board with originally, which was using a dark theme

QUICK GUIDE
What is KDE?

Many long-standing Linux users associate the KDE project as a whole with the desktop environment, because of the project's origin as Kool Desktop Environment. This name for the desktop was later abbreviated to KDE and in 2009, the project shifted away from its desktop focus and now sees 'KDE' as the name for the community. This community is made up of hundreds of people who work on products broken into three key parts (detailed, right).

KDE Plasma
The primary user interface which is optimised for desktops and laptops. A UI for mobile devices called Plasma Mobile is in development (See LU&D195, p58)
KDE Framework
A set of high-level libraries for UI development and system integration. The framework can be run anywhere, including on GNOME and even Windows.
KDE Applications
This includes a wide array of applications from file managers (Dolphin) and text editors to areas such as personal finance (KMyMoney).

image credit: Konqi mascot art: Tyson Tan CC-BY-SA

SPOTLIGHT
Paying for KDE

When discussing GNOME and KDE's "friendly competition", we asked Graham whether there was a feeling within the community that KDE could benefit from a commercial arm, or a company that supports them more significantly in financial terms. "This is a very controversial question, and it's something that we're actually in the middle of debating right now – this question of when, where, how and who should pay developers," he says.

KDE does have Blue Systems, which sponsors quite a lot of work and employs many core Plasma developers, but as Graham admits, "We don't have a Red Hat behind us." There are other companies involved. "Our flagship digital art product, Krita, is backed by the Krita Foundation," he says, and the foundation collects donations and pays developers. "I think there are already moves in that direction. I don't want to try to influence the result of this internal discussion, but needless to say, it's something that we're thinking a lot about in the community."

His personal view is that paid development doesn't seem to be impacting the project negatively right now, but, he says, there's some controversy to the KDE e.V. Foundation – the non-profit association that legally represents KDE – paying developers, which is something that the community is discussing. But he doesn't think there are many objections to more companies contributing.

Above Volunteer power: KDE exists on a small budget, which in 2016 was 98,160 euros (when you remove QtCon from the KDE e.V Foundation figures, which cost more to put on than it brings in). These figure don't include contributors who are paid by companies to work on KDE, but does include foundation staff costs, which are only 18,230 euros

for Plasma and a light theme for apps. That was something I fought against internally, but now it's been released, people really seem to like it. So I'm going to say: I was wrong, you were right – that was a great decision. That's one of the things that I think makes our community really strong. We have a lot of people who are interested in pushing the envelope.

What do you feel were the key developments from Akademy this year?
That's a really great question. I think there were a couple of things that were really important. First of all, the Kate and KTextEditor developers did probably a year's worth of work in one week, which was astonishingly amazing. It was an incredible pace of improvement. Other than that, there was lots of work on Plasma, too. There were lots of really great conversations that we all had regarding topics that are sometimes hard to discuss over the internet but are a little bit easier to have face to face. There isn't really anything specific that I'd like to talk about just yet, but we got rid of a lot of bottlenecks.

So for developers interested in getting involved, what kind of support does the KDE project offer?
We have a wiki page that's linked from the main page on KDE.org. Just click 'Get involved' at the very top, and it takes you to a whole list of things that you can do. We've got detailed documentation and instructions for how to file good bugs, how to triage bugs, how to get involved in development, how to get involved in visual design and translation. We're always working to improve that documentation, but we think it's at a pretty decent stage right now.

Once you decide that you want to get started, there are a lot of great resources for communicating with developers. We've got IRC channels, we've got Telegram rooms, we've got mailing lists and then of course, our system for code review and test management – Phabricator [**https://phabricator.kde.org**] is somewhere where we're all very active.

On privacy, what developments are happening?
The privacy side is not as active as the other two [goals], but there are definitely some things going on behind the scenes. There's been a lot of preparation for work that hasn't been made public yet.

Is that related to Purism's Librem 5 phone?
I think it's related, in the sense that all of these things are kind of related, because we're doing lots of work with them. I think when it comes to the Librem 5 phone, we want to make our platform more attractive for that target. I think one of the ways we're doing that is with the GNU Ring [a universal communication platform focused on protecting privacy and freedoms]. This is the KDE client for the GNU Ring software. We've just released a new version recently that uses our Kirigami framework in order to improve the user interface. So we've just got to keep going with that, and we really hope that we can strengthen our partnership with [Purism], and eventually convince them to use Plasma Mobile.

We were under the impression that Purism wanted to support it?
My impression is that [Purism] are, right now, going with a custom-made GNOME solution. But they're committing to supporting a community-maintained KDE Plasma version. So by default, I don't believe they're currently planning to ship with Plasma Mobile. That's something we hope to change.

> "We've gotten a very good reception from Kubuntu 18.04– users are pretty happy with it"

OPINION

The kernel column

The legendary Jon Masters' final column, where he summarises the latest happenings in the Linux kernel community

Jon Masters
Jon is a Linux-kernel hacker who has been working on Linux for more than 22 years, since he first attended university at the age of 13. Jon lives in Cambridge, Massachusetts, and works for a large enterprise Linux vendor, where he is driving the creation of standards for energy-efficient Arm-powered servers.

Linus Torvalds announced the release of Linux 4.19-rc1 (Release Candidate 1),and with it the closure of the 4.19 merge window. The merge window is the period of time (almost always two weeks long) during which disruptive changes and new features are allowed into the kernel. This is followed by a number ofweekly RCs and a final release about two months later. The latest merge window was "fairly frustrating" to Linus because "4.19 looks [to him] to be a pretty big release", and it came right as a new hardware security vulnerability also needed to be patched (more on that below). Linus also drew attention to fixes for a nasty TLB shootdown bug (also covered below) that consumed his time as he was handling other merge window activity. He added that, "while this isn't the biggest release we've had (4.9 still keeps that crown), this does join 4.12 and 4.15 as one of the bigger kernel releases". He later announced a follow-up RC2, which contained mostly networking and GPU driver fixes.

L1 terminal fault explained

Another month, another vulnerability disclosure impacting the 'speculative' design of modern microprocessors. As we've mentioned in the past, contemporary high performance processors (such as those in your laptop, or in a server) try to predict future work that they will need to perform before they know which direction a program will take in its operation.

This is known as speculation and its supposed to behave as a black box: either the work done ends up being needed, and becomes part of the 'architectural state' of the machine, or it is thrown away and its results are supposed to not be visible to programmers or users. Since this is the intent, various additional optimisations are common in which the processor will also speculate that certain other things are true, such as the result of a permission check (as in Meltdown).

As Meltdown demonstrated, if the speculative activity can be (indirectly) observed, certain other performance optimisations relying upon speculation become security issues, potentially allowing unprivileged users to gain access to sensitive information. The latest problem concerns a design optimisation Intel made to its handling of page tables. These are used by a part of the processor known as the Memory Management Unit (MMU) to handle the mapping of programmer-visible virtual addresses into underlying hardware physical addresses. The abstraction of virtual memory is what allows every program to think it has unlimited amounts of RAM, with the kernel being able to swap in and out memory from disk whenever the amount of free RAM is low. The kernel marks memory pages that are swapped as invalid ('not present').

Every running task or process has its own set of page tables, as does the kernel itself. Whenever virtualisation is in use, the Hypervisor (KVM, say) also uses its own page tables to translate what the guest virtual machine sees as its memory into real underlying memory in the host. Thus, two separate sets of translations may be required. Page table walks, as they are known, can be slow, so the processor uses various optimisations to speed them up, including a small internal cache of recent translations known as a TLB (Translation Lookaside Buffer). Even then, walking through page tables is so slow that the processor might speculate about the result ahead of time.

In Intel's case, it speculates that the page table entry (the leaf node in a page table walk) is valid before completing the validity check. This allows carefully crafted applications to cause the creation of malicious 'not present' page table entries (PTEs) that look to Linux just like swapped-out memory. Various forms of attack exist, with the most dangerous relying upon a second bug that affects virtualisation. When the hardware sees a 'not present' page table entry, it will skip the second page-table walk in the hypervisor and treat the partially translated address as being a hypervisor physical memory address, allowing a specially crafted malicious VM to read hypervisor memory.

Pulling off an attack is currently difficult, since ready exploits are not yet in the wild, and relies upon causing data of interest to also be loaded into the processor's level 1 data cache (L1D). Patched kernels carefully avoid accidentally creating malicious 'not present' pages, and mitigate against the hypervisor attack by flushing the L1 data cache whenever beginning to run potentially untrusted virtual machines.

This can be controlled using a new 'l1tf' kernel command line, and the status of any vulnerability and possible active mitigation is visible in `/sys/devices/system/cpu/vulnerabilities/l1tf`. Linus mentioned that a nasty TLB shootdown bug had

been found during the 4.19 merge window. As we learned above, TLBs are used to cache the translations between virtual addresses used by applications (or the kernel) and those seen by the physical hardware. Since every running application has its own (virtual) view of memory, these TLB entries need to be maintained such that only those translations that are supposed to be active are seen by an application accessing memory. Otherwise, it would be possible for a memory access by one application to interfere with another, or for it to see a stale translation into memory that was since recycled for another use.

As a consequence, the kernel routinely performs 'shootdowns' of TLB entries as it performs page table maintenance operations. The exact process differs from one architecture to another, but usually involves specially privileged processor instructions, as is the case on x86. Unfortunately, some time ago, an attempt was made to optimise the process of tearing down user process page tables and the result was that x86 might – in some cases – not do the necessary TLB flush. This resulted in a very hard to debug issue that also had security implications.

> "The basic idea of the Energy Aware Scheduler is to prefer to schedule newly awaking tasks onto energy-efficient cores"

It's interesting to see how the change had originally occurred through an attempt to make PowerPC-specific code generic across architectures. Linus was grumpy because the net result had been that "x86 had unintentionally lost the TLB flush we used to have". The problem had then existed for a long time because it only affected a rarely used 'slow path' – triggering an exploit required that the machine intentionally ran out of memory. The thread is titled "Remove tlb_remove_table() non-concurrent condition", and any theoretical security impact is also fixed in the updated patches.

Meanwhile, Pu Wen (of Hygon) posted several rounds of updated patches for the Hygon Dhyana family of x86 processors. These are based upon AMD's Zen microarchitecture as a result of a joint venture between AMD and a Chinese group known as Chengdu Haiguang IC Design Co. – Hygon. Dhyana appears to use AMD's Zen cores on a custom SoC (chip), so much so that the only real changes needed to enable support for these chips in Linux is adding some new IDs, and replacing CPUID detection of 'AuthenticAMD' with 'HygonGenuine'.

Atish Patra (of Western Digital) posted various clean-ups to the SMP (Symmetric Multi-Processing) supporting code for RISC-V architecture. These patches focus on one implementation (from SiFive), which is believed to be used in some future Western Digital disk drives, but more generally are good examples of the rate of progress. Apparently, the work was inspired by ARM's SMP code.

Quentin Perret (of ARM) posted version 6 of a long-running patch series titled Energy Aware Scheduling. The EAS (Energy Aware Scheduler) attempts to divide a machine into performance domains built from various components, such as processor cores. The energy requirements of these different performance domains vary, as does the amount of computation they can provide. The basic idea of the EAS, then, is to prefer to schedule newly awaking tasks onto energy-efficient cores, and to migrate them onto higher performance, higher energy cores when needed. Various logic in the patch set ensures that once a 'tipping point' is reached and the machine is sufficiently loaded, it will fall back to the traditional scheduling philosophy in use today. EAS is not specific to ARM and in fact it is hoped it will be supported across many different architectures. Once enabled, `/sys/devices/system/cpu` gains an 'energy_model' directory showing the energy cost and relative performance of the various performance domains.

OPTIMISE LINUX

Use Linux's famed malleability and streamline it to power everything from old computers to IoT devices – **Mayank Sharma** explains how

AT A GLANCE

Light distributions p20
Everything you need to make your trusty old workhorse usable again.

Improved performance p22
Extract every gram of performance from your everyday desktop.

Custom minimal distribution p24
Ditch ready-made solutions and craft your own lean Linux by hand.

Exploit the Pi p26
Make the best use of the hardware upgrades in the popular little powerhouse.

Build an IoT device p28
Use the Pi to turn appliances into Internet of Things-enabled smart devices.

Specialised distributions p29
Tuned distributions with a purpose designed for a solitary task.

Straight out of the box, the typical Linux desktop distribution is a fairly well-oiled machine. The developers at every stage of the distribution assembly line, from the kernel upwards to the individual apps, put in a huge amount of effort to ensure that the different pieces of software make best use of available hardware resources. But the mainstream distributions have to appeal to a wide audience and ensure compatibility with a range of hardware. The downside to this approach is that most popular distributions aren't optimised for your specific hardware.

This leaves plenty of wiggle room for optimised solutions, which are tailor-made for specific applications, devices and use-cases. This feature is all about such fine-tuned, purpose-made distributions, and over the next few pages we'll help you find one – whether you want it to run your old workhorse or your IoT-enabled smart home.

Perhaps the most well-known application for lean Linux distributions is to revive ageing hardware wasting away in the attic. The pace of hardware developments combined with the voracious appetite of modern resource-guzzling apps is rendering hardware obsolete almost as soon as you take it off the shelf. Distributions that bring these back to life are of use to a large number of users.

"Perhaps the most well-known application for lean Linux distributions is to revive ageing hardware wasting away in the attic"

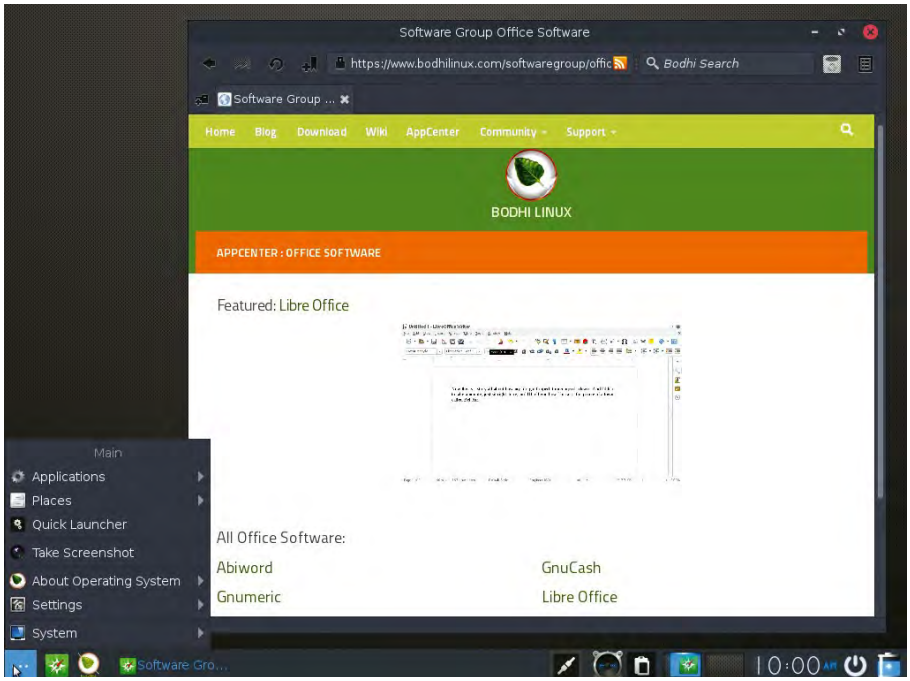
Also in vogue are distributions made for a particular type or class of hardware, such as netbooks and Chromebooks, as well as renaissance devices like the Raspberry Pi. ChromeOS worked well for the first generation of Chromebooks; it was responsive and did what it was designed for. But it doesn't do justice to the newer, more powerful Chromebooks. We'll help you find solutions that enable you to squeeze every last bit of performance from your Chromebooks to deliver an experience reminiscent of a traditional desktop Linux.

One casualty of the rise of the Chromebooks are netbooks. The netbook fad evaporated before vendors could ship devices with adequate resources to power the newest line of software. If you're still holding on to one of these underpowered laptop-wannabes, we'll help you put them back into active use, with distributions that will chug along nicely in these hardware-limited environments.

Another device that wasn't conceived as an everyday desktop is the Raspberry Pi. The ARM-based device is a perfect educational computer, of course, but it can be a lot more than that. There are distributions that'll turn the puny little device into an optimised server or help you build smart IoT devices, and we'll look at some of these too.

Light distributions

Don't have the heart to scrap your trusty old workhorse? Here's everything you need to make it adorn your desk again



Left Bodhi's default installation doesn't have many apps and you'll have to pull in the ones you need using Bodhi's browser-based Appcenter

You won't find many people on the planet who still use EISA FDDI adaptors, which were phased out in the '90s. Maciej W. Rozycki is one of the few who does, and he's the only reason why the Linux kernel continues to support this ancient piece of tech. While the kernel can afford the luxury of supporting such legacy hardware as long as it has even a single user, most distributions and individual open source projects cannot.

Forget about decades-old hardware; most distributions don't think it's viable to even support 32-bit architecture any more. Everything from new user projects such as Solus and Elementary OS to the privacy-centric Tails Linux has dropped support for the 32-bit platform. Even mainstream projects like Arch and Ubuntu no longer spin installation ISOs for 32-bit machines.

Despite the fact that these machines have been missing from the shelves for quite a while now, they still adorn a lot of desks all over the world. Thankfully, you can still put these old machines to good use – and it really isn't a surprise that a large number of open source developers are working hard to make obsolete hardware usable again.

Revive old computers

Labelling hardware as 'older' is tricky. The pace of hardware development is rendering even relatively newer hardware obsolete. Examples of these relatively recent attic-ready hardware would be a single-core or dual-core AMD Athlon/Intel Pentium with a couple of gigabytes of RAM that it shares with onboard graphics. In the not too distant past, mainstream distros would perform adequately on these machines, but no longer; nowadays, you need special-purpose distros to make good use of such hardware.

Tiny Core (**http://tinycorelinux.net**) will always be our first choice for powering computers that are light on resources. Its original developer, Robert Shingledecker, was previously involved with the once-popular but now-dormant Damn Small Linux project. The distribution is available in three flavours. There's a miniscule 11MB Core

QUICK GUIDE

Raspberry Pi for (ageing) desktops

Above The Raspberry Pi Desktop can be used from a USB disk with persistence mode

The Raspberry Pi made its foray onto PCs in late 2016. The intention was to enable users to experience their recently introduced desktop environment on regular x86 computers. The Raspberry Pi Desktop has modest requirements, which makes the distribution a viable option for resurrecting underpowered computers.

A positive side effect of using the Debian base is that while on the Pi the software repository is limited to packages compiled for ARM chips, on the desktop you have access to a much wider collection of programs. The desktop flavour of the distribution contains all the software you'll find on the Raspberry Pi version. Additionally, there's the Chromium web browser, Claws mail client and the LibreOffice suite. You can also use the PiPackages package manager to easily flesh out your installation.

edition that'll help custom desktops, while the recommended TinyCore edition weighs in at 16MB and boots to a graphical desktop. At the top end is the 106MB CorePlus edition that has additional drivers for wireless cards, a remastering tool and localisation support. The oldest computer we've managed to run this heavier edition on was a 233MHz Pentium II box with just 64MB of RAM.

The hardware requirements of the TinyCore edition is even lower. True to its name, this version bundles just a terminal, a text editor and an app launcher on top of the lightweight FLWM window manager. It has a control panel to manage boot services and configure the launcher. If you need anything else, you'll have to pull it in using its package manager, including the installer if you want to install the distribution. Due to its minuscule size, TinyCore boots blisteringly fast.

> "Examples of relatively recent attic-ready hardware would be a single-core or dual-core AMD Athlon/Intel Pentium with 2GB of RAM"

Remember however that the distribution is designed for advanced users, and you'll need to spend some time fleshing it out using its package manager, which isn't the most intuitive in the business. You'll also have to browse through its documentation to familiarise yourself with the distribution's peculiarities, irrespective of your experience with Linux.

Revive Chromebooks & netbooks

Both Chromebooks and netbooks were designed to replace traditional laptops. While Chromebooks are essentially a cloud-based device, netbooks could run traditional desktop operating systems. The novelty of both devices has worn off over time. With Chromebooks this was due to the lack of flexibility of Chrome OS compared to a traditional distribution and netbooks simply couldn't keep pace with the hardware requirements of the latest software. In the long run neither has been able to emulate the usability of the traditional portable computer they were pitched to replace; Chromebooks are limited by their operating system and netbooks by their hardware. The good news is that there are options to help you make these stylised paperweights usable again.

GalliumOS (**https://galliumos.org**) is your best bet to increase the usability of Chromebooks. The lightweight distribution has several optimised versions for the different CPU architectures of the Chromebooks. It also features some other optimisations to squeeze performance, such as the zram kernel module to create a compressed swap space in RAM and the BFS and BFQ kernel schedulers to increase stability. The developers have also taken some steps to increase the battery life, and increased boot speed by using a smaller kernel and fewer number of services. Before downloading it, make sure you check the Hardware Compatibility list of supported models for any device-specific instructions.

Similarly, you can use Bodhi Linux's Legacy edition (**www.bodhilinux.com**) to press a netbook back into active duty. This edition of the distribution caters to 32-bit computers and uses an older kernel optimised for hardware older than 15 years. Bodhi uses its own home-brewed Moksha-desktop which is forked from the Enlightenment 17 desktop. Moksha paints an elegant desktop with subtle animations and effects without consuming too many resources. The same is true for the distributions' default apps. The Midori web browser, the ePad text editor, the PCManFM file manager and others are all lightweight alternatives that support virtually all commonly used functions of their more resource-hungry mainstream cousins.

The combination of the optimised non-PAE kernel pulled from the upstream Debian repositories, along with the lightweight components, help Bodhi Legacy perform well on Atom-based netbooks. In our tests, it worked wonderfully well on an 1.8GHz Atom N475 processor with 2GB of RAM. The official minimal requirements of the distribution are even less. The developers recommend a 1GHz processor with 512GB of RAM though you'll want 2GB for optimal performance.

EXPERT TIP

Jeff Hoogland
Lead developer, Bodhi Linux

As far as building our ISO images go, we generally start from an Ubuntu LTS mini ISO image and then add what we need for a base system from there. Our goal is to get as few packages as possible installed, while still providing the user with a GUI to work from. A couple of members of our team maintain a fork of the old remastersys script, which we've branded 'bodhibuilder', that we use to spin up the actual ISO images.

QUICK GUIDE

Performance-enhanced kernel

Besides the stock kernel there are several performance-optimised kernel patches available on the internet. One of the most popular is made by Linux kernel developer Con Kolivas, who assembles it from lots of different performance patches.

To install it, use your distro's package manager to first download the kernel source for the version the patch applies to, along with all the tools for building your kernel.

Then download the patch from Koliva's website (**http://users.on.net/ckolivas/kernel**) and use the patch command in the kernel source folder to apply the changes, such as `sudo patch -p1 < patch-4.*-ck1`. Then just follow your distribution's instructions to build the kernel.

Ubuntu users can try to automate the process by downloading and modifying the script at **http://bit.ly/lud_kernel** which will download a vanilla kernel along with Kolivas' patches and compile them into installable binaries for you.

Improve performance

Lean, optimised distributions aren't just meant for reviving old hardware – they also make sense on newer, modern computers

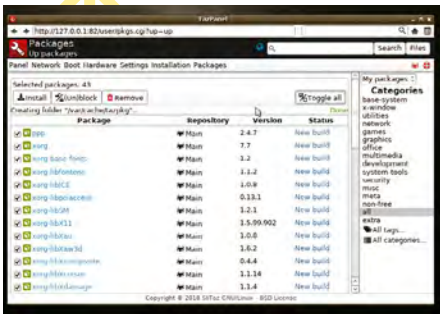
Earlier we mentioned that mainstream distributions are all about coverage and range. They are designed to be useful to a large number of people. The downside to this approach is that they're bloated with lots of unnecessary components that might make them lethargic and slow on your hardware. This is exactly where slimmed-down distributions help you make best use of the available resources.

Get a ridiculously fast desktop

One of the best options for accelerating regular desktop Linux tasks is Linux Lite (**www.linuxliteos.com**). The latest version is built using packages from the Ubuntu 18.04 release and uses the Xfce desktop with aesthetically pleasing modifications. The distribution offers a good compromise between speed and functionality, and exposes all useful desktop features while still being responsive and fast. The distribution will work blazingly well on anything with a processor faster than 1.5GHz and 1GB or more of RAM.

Linux Lite looks and feels like a regular mainstream distribution. Its list of pre-installed programs doesn't include any of the traditional lightweight apps and is instead brimming with the usual suspects such as GIMP, Firefox, VLC and LibreOffice. On top of the this you can also easily pull in other popular programs such as Kodi, Skype, Steam, Spotify and more using the custom Lite Software application. In fact, the distribution includes several homebrewed Lite-branded tools for interacting and customising various aspects of the desktop and the installation.

There's a welcome screen that enables users to install updates and drivers, and set up a backup restore point, a program to help upgrade to the latest release, and another to manage users. One of the most useful ones is Lite Tweaks, which helps users with common admin tasks. You can use it to free up memory, remove older kernels and install new ones. It also includes some performance-enhancing tweaks such as installing and enabling zRAM and Preload.



Above SliTaz is optimised to live in RAM and is easy to maintain since it's a rolling release

Another such lean distribution is MX Linux (**https://mxlinux.org**), which is based on Debian Stable and is a joint effort between the antiX and MEPIS communities. Its clean,

QUICK GUIDE

Build your installation from scratch

One approach towards a lean Linux installation is to do it yourself. Two of the most popular options for building a streamlined Linux installation are Gentoo (**www.gentoo.org**) and Arch Linux (**www.archlinux.org**). Both distributions are available as minimal netinstall ISO images that help you put together a fully functional system one component at a time. The big difference between the two is that Arch officially now only supports 64-bit machines. The Lego-esque installation approach of both the distributions translates into a lengthy and involved process when compared to the other options mentioned in these pages. It'll also take up more time and eat a fair amount of bandwidth as well.

On the upside, though, you'll end up with a far more fine-tuned and streamlined installation.

Another source-based option for advanced Linux users is Crux (**https://crux.nu**), which inspired the creation of Arch. It too is designed for users who are ready to give away the usual conveniences such as a graphical configurator, or even an installer, in order to fine-tune their installation to their heart's content. All three distributions have very detailed handbooks that will handhold you through the entire process of crafting your installation.

If you're really committed to the idea of a streamlined Linux, you need Linux From Scratch (**www.linuxfromscratch.org**).

Above Linux From Scratch has spawned various sub-projects like Beyond LFS to flesh out the LFS system, and Automated LFS to automate the process of creating an LFS system

EXPERT TIP

Prepare to update

MX Linux updates a lot of apps and back-ports newer versions from Debian Testing – so you'll have to do a fresh install every time the distro switches to a new Debian Stable release.

iconless Xfce desktop displays basic system information via an attractive Conky display. The USP of MX Linux is its user-friendliness and an array of customisation tools. Some of its noteworthy MX-branded tools are a snapshot tool for making bootable ISO images of the working installation, and a remaster tool to create your own customised version of the distribution. The desktop boots to a welcome screen that contains useful links to common tweaks and the distribution's custom set of tools. Like Linux Lite, MX Linux comes with a host of mainstream apps, many of which have been back-ported from Debian Testing.

Run entirely from RAM

Another way to speed up a contemporary computer is to run the operating system entirely from its RAM, which has a much higher I/O rate than a hard disk. Once the distribution has copied itself to memory, app launches are several times faster than with a traditional hard disk installation. The real trick here, however, is to find such a distribution that's optimised to live in RAM yet which still provides you with a complete desktop environment.

> "Another way to speed up a contemporary computer is to run the operating system from RAM, with a much higher I/O rate than a disk"

One such distribution is SliTaz (**www.slitaz.org**), which has been chugging along for over a decade now. The rolling release distribution has a stable and a developmental version. It takes up less than 100MB RAM and yet offers a graphical desktop (along with some optional desktop effects), includes web browsers, audio and video players, several code editors and more, and you can install others using its graphical package manager. The distribution also has a bunch of custom tools such as SliTazPanel with which you can administer all aspects of the system. SliTaz is built with home-brewed tools known as cookutils and uses BusyBox for many of its core functions. You can also anchor SliTaz to your hard disk and Windows users can host it inside a directory without partitioning their disks. The distribution is available for both 32-bit and 64-bit platforms. If you aren't averse to reading through some documentation and learning new tools, SliTaz is an excellent option for powering a computer from RAM.

Another esoteric option that's again optimised to run entirely from RAM is AUSTRUMI (**http://cyti.latgola.lv/ruuni**). This Slackware-based distribution is only available for 64-bit computers. Unlike SliTaz, AUSTRUMI's boot menu gives you the option to eject the Live medium after it has copied itself to RAM. There's also another option to keep the Live medium in place, which helps the distribution perform on machines with less than 1GB of RAM.

AUSTRUMI doesn't use a full-blown desktop environment and instead relies on a customised instance of the FVWM window manager that offers several tweakable options. The distribution uses an automatic wallpaper switcher and ships with several different themes. Unlike its RAM-only lean peers, AUSTRUMI's live environment includes regular productivity apps such as Firefox, GIMP and even LibreOffice.

Lightweight alternatives

Looking for less hardware-hungry alternatives to your usual Linux apps? Try these

Component	Category	What it offers
LXQt	Desktop	Combines LXDE components for an ultra-fast desktop with a modern Qt look and feel
IceWM	Window Manager	Highly malleable standards-compliant standalone window manager
Ly	Display Manager	ncurses-based app based on Linux-PAM and Systemd
PCManFM	File manager	LXDE's default file handler is lightweight and does most important tasks
Krusader	File manager	A graphical orthodox file manager based on the venerable Midnight Commander
Midori	Web browser	Built on webkit, it is reasonably compatible with the modern web
Trojitá	Email client	Robust IMAP-only client that has the usual features minus the bloat
MPV	Media Player	Spiffy looking player based on mplayer2 code
Feh	Image Viewer	A simple app for console users with a useful set of features
Psi	Chat Client	Supports the XMPP/Jabber protocol and bundles the usual IM features

Purpose-built distributions

There are a plethora of lean distributions that have been pruned and tuned for a particular reason – try these

Trimmed distributions aren't only designed for ageing hardware – we've already seen distributions that are built to perform blazingly fast on a modern computer. Similarly, there are several others that are built with a particular optimisation or application area in mind.

Optimised for USB

Porteus (**http://porteus.org**) is a portable distribution designed and optimised to run from removable media such as a USB flash drive, SD card or CD. If the medium is writable, it'll save all changes inside a folder and load them on subsequent boots. The distribution is based on Slackware Linux using a modified version of the linux-live scripts that help cut down boot-up and shutdown times.

The latest release supports newer EFI machines and also includes the latest Intel firmware fix for the Spectre vulnerability. Instead of a single ISO, Porteus is available as seven separate ISO images, each with a different desktop – from the heavyweight KDE and Cinnamon to the lightweight LXDE and Openbox. Surprisingly, the ISO of most editions weigh in around 300MB each. This is because the distribution exists in a compressed state on the storage media and the familiar Linux directory structure is created on the fly during boot. Another unique aspect of the distribution is its use of modules, which are pre-compiled packages that you activate and deactivate as per your requirements. When you fetch packages via Porteus' package manager, you also get the option to convert them into modules. You can then activate the modules to use the app. If you move these app modules to a designated directory inside the removable storage, the app is available in subsequent boots as well.

Going without Systemd

Although a majority of distributions have shifted to the Systemd system and service manager, its opponents believe it adds a level of complexity that goes against the UNIX philosophy. Several distributions have weeded out Systemd and continue with the SysV init system.

antiX (**https://antixlinux.com**) is one such dissident. In addition to sticking to SysV, the distribution also prides itself on its Live USB skills. The distribution has a unique customisable boot loader that enables you to save custom boot entries. Also of note is the live-usb-maker tool that enable features such as persistence, live remaster and several others. antiX offers a choice of lightweight desktops and window managers and even offers a core edition that ships with just enough applications to help you build your desktop from scratch.

The full edition of antiX is one of the lightest distributions around thanks to its choice of IceWM together with the Rox file manager for the desktop along with a very well curated collection of apps. A handful of mainstream apps, such as Firefox and LibreOffice, are complemented by

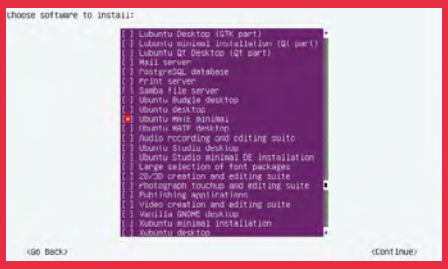
EXPERT TIP

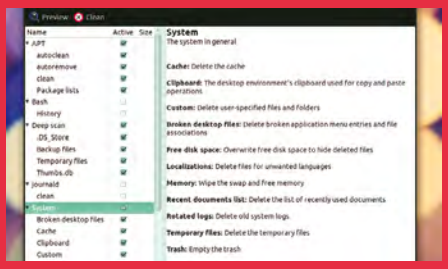
Porteus Kiosk

This is a special lean version of Porteus designed for use on a publically accessible computer. You can set it up as a diskless workstation or for digital signage using its setup wizard.

HOW TO

Create your own minimal distribution





1 Install minimal Ubuntu

Begin by downloading the minimal Ubuntu ISO that includes the least number of components required to boot Linux. During installation, the installer shows the tasksel tool for selecting additional components; select one of the minimal graphical environments.

2 Customise the install

After installation, boot into the distribution and add and remove packages as per your needs using the package manager. You can also manually install apps by downloading pre-packaged binaries and modify other aspects of the installation such as menus and themes.

3 Clean the install

When you've made all the changes, it's best to remove any unnecessary files from the installation. While you can do this manually, save yourself a lot of time by using BleachBit. Also head to Edit > Preferences > Languages to remove any additional language packs.

QUICK GUIDE

A special woof-out to Puppy Linux

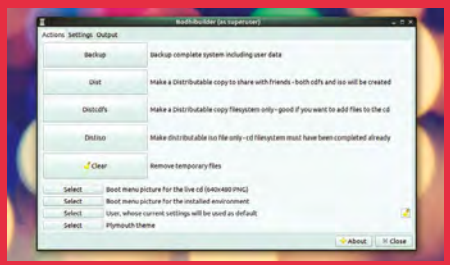
One of our all time favourite tailor-made distribution is Puppy Linux (**http://puppylinux.com**). The project actually has a couple of official releases along with some from its community of users thanks to its well-documented build scripts. Of its two official releases, one is based on packages from Ubuntu Xenial and the other on Slackware 14.1. Like antiX, Puppy saves

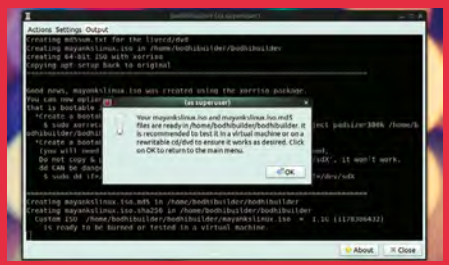
> “Several distributions have weeded out Systemd and continue with the SysV init system”

lightweights such as GNOME media player, Claws email and Dillo web browser along with more esoteric but useful apps like the Droopy network file-sharing web server. The distribution also uses a handful of custom tools such as the package installer and the antiX control panel, which enables you to modify aspects of the distribution.

Running webapps

Right Unlike the usual slew of lightweight distributions, AntiX is only lean in terms of its resource utilisation





4 Install bodhibuilder

Bodhibuilder is a graphical tool that has been forked from the remastersys script by the Bodhi Linux project; we'll use it to turn our custom installation into a distributable ISO image. The script is available as a binary and works on all Ubuntu-based distributions.

5 Set parameters

Fire up the app and switch to the Settings tab to define the name of your custom distribution. The Actions tab offers several build options. The Backup option will roll the installation into an ISO image with the exact same settings as your desktop and is meant for your own use.

6 Build and share

To create a distributable ISO, you should use the Dist option. This excludes all your personal settings and only includes your app selection. Before you press this button, you can optionally use the buttons at the bottom of the interface to set a custom image for the bootloader.

Exploit the Pi

The Raspberry Pi was created to be a low-cost educational computer, but thanks to hardware upgrades it can be a lot more

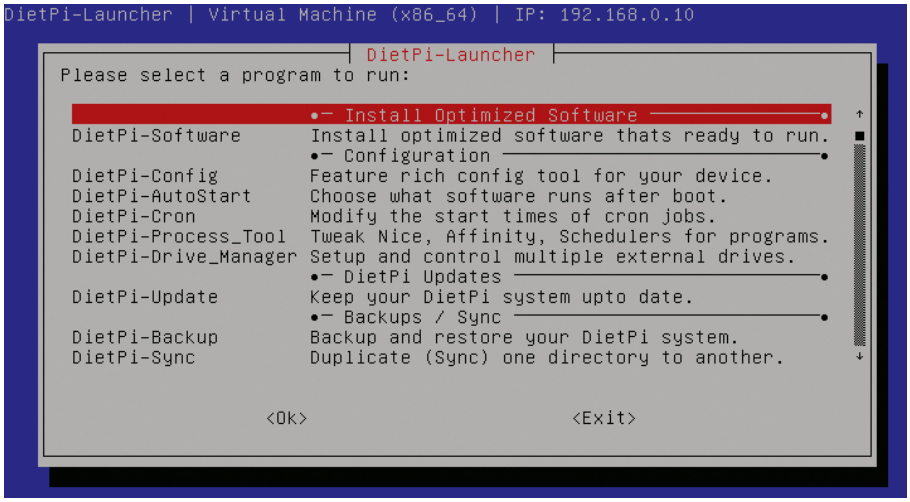
The Raspberry Pi was designed as an affordable and functional computing device for kids who wanted to learn to program. However, the device hit it off with the hackers and modders who began using it creatively and made it usable to audiences and applications that the original creators hadn't imagined. The latest iteration of the inexpensive computer has ample resources and yet is small enough to run everything from your smart lamp to your file server. While Raspbian does an excellent job as a general-purpose distribution for the Pi, specialised applications require specialised distributions, such as the following.

Optimised deployments

One of the most popular uses for single-board computers such as the Raspberry Pi is as an always-on and efficient standalone server. DietPi (**https://dietpi.com**) is one of the best options for rolling out such a server.

DietPi installs the bare minimum components you need to flesh out the installation according to your requirements. The reason behind the distribution's success is its handful of custom scripts, one of which enables you to install software optimised for the Pi. Server distributions are notoriously difficult to set up, but DietPi simplifies the process by offering a nice menu to help you pick and choose a functionality for your device. For example, using DietPi's custom package management script you can turn the base installation into a streaming media server, file sharing server, backup server, web server, VPN server, seed box and lots more. If servers aren't your thing, DietPi gives you the option to turn it into a graphical desktop with either Mate, LXDE, Xfce or GNUStep desktops. As it's a Debian-based system, outside of the script you can use the command-line APT package management system, or pull-in the Synaptic graphical apps for easier package management. Furthermore, the custom script also enables you to overclock the Pi, mount remote shares and even enable the Pi camera module.

> "Server distributions are notoriously difficult to set up, but DietPi simplifies the process with a menu"


DietPi-Launcher | Virtual Machine (x86_64) | IP: 192.168.0.10
DietPi-Launcher
Please select a program to run:
•— Install Optimized Software
DietPi-Software Install optimized software thats ready to run.
•— Configuration
DietPi-Config Feature rich config tool for your device.
DietPi-AutoStart Choose what software runs after boot.
DietPi-Cron Modify the start times of cron jobs.
DietPi-Process_Tool Tweak Nice, Affinity, Schedulers for programs.
DietPi-Drive_Manager Setup and control multiple external drives.
•— DietPi Updates
DietPi-Update Keep your DietPi system upto date.
•— Backups / Sync
DietPi-Backup Backup and restore your DietPi system.
DietPi-Sync Duplicate (Sync) one directory to another.
<Ok>
<Exit>

Above Besides the Raspberry Pi, you can run the DietPi distrbution on several other SBCs including BeagleBone, Banana Pi, Orange Pi and more

Docker in production

Before you can use Docker in production you need to decide an OS to run it on. You can run it on just about every Linux distribution, but RancherOS (**https://rancher.com/rancher-os**) gives you a minimalist Linux environment that's tailored to explicitly run Docker. RancherOS is stripped right back to the bare essentials with only the minimum amount of software needed to run Docker. Everything inside the OS runs as a Docker container. Architecturally speaking, there's the Linux kernel over which runs a system docker as the first process, which manages all the system services such as rsyslog and udev. Then there's another container that runs all the other containers that users want to run. This arrangement works well for a production server as it creates a nice level of isolation and avoids system services being impacted by the users.

Although the OS itself is very interesting, what really makes it useful is the Rancher platform that runs on top of it. Rancher can manage and monitor container hosts and provides things like resource management, container networking, service discovery via DNS, load balancing based on HAproxy and more. You can run RancherOS on bare metal, virtual hosts and various cloud platforms. One of the bare-metal platforms it supports is the Raspberry Pi 3. Transfer the image to an SD card and refer to the online documentation to get started.

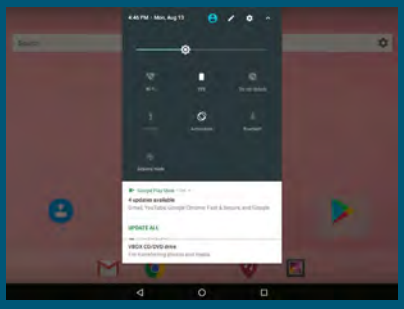
Power IoT devices

Besides education, the one area in which the Raspberry Pi has had a monumental impact

QUICK GUIDE

Android for the desktop

While the idea of running an operating system designed for mobile platforms on top of regular-sized computers sounds cumbersome, Android's low resource requirements makes it ideal for older hardware. Android-x86 started off as a series of patches to the vanilla open source Android code to get it working on netbooks such as the ASUS Eee PC, but has grown into a fully fledged distro that supports both 32-bit and 64-bit machines. The project currently has two stable releases, one based on Android Nougat and the other on LineageOS (earlier known as Cyanogenmod). The releases are available as Live installable ISOs which helps you to check whether the distribution supports the peripherals on your old computer. In our tests, the distribution picked up all the hardware including the touchscreen on an old Acer 5738PZG. At the end of the day, Android-x86 doesn't offer the same flexibility as a full-fledged desktop distribution, but is an attractive proposition to make an old defunct machine usable again.



Above Use RPM files to install Android-x86 inside a folder in the root directory, along with relevant boot options in GRUB

EXPERT TIP

Docker distributions

There are several other lean Linux distributions for running containers, each with their own advantage. Besides RancherOS, there's ContainerOS (which uses Kubernetes), RedHat's Atomic Host (ideal if you run Red Hat compatible servers), and Alpine Linux (used by Docker to build its packages).

is the Internet of Things (IoT). The open, hackable nature of the small and inexpensive computing device has made it one of the most popular prototyping and development boards for developing and programming all kinds of IoT projects.

Of course, standard bare-bones distributions such as Ubuntu Core (**www.ubuntu.com/core**) give you a wonderful base platform to code your IoT projects – but you don't really need to be a proficient programmer to interact with your appliances and devices. You can use Prota Pi (**https://prota.info/prota/pi**) to make the Pi talk to your devices without writing any code.

> "You don't really need to be a proficient programmer to interact with your IoT devices"

Prota Pi is a headless OS with which you can interact with the devices from a point-and-click, HTML5-based user interface. The core automation engine of Prota OS is named Stories and helps you create automation rules for connected devices. These rules are based on the apps available in Prota Pi. One of the most useful is the GPIO app that allows you to control all the GPIO pins on a Raspberry Pi from directly within Prota Pi's remote interface.

Check out some of the interesting IoT and automation projects shared by the developers of the Prota Pi, Naran Inc, on instructables.com (**www.instructables.com/member/Naran/instructables**) to get a taste of the possibilities on offer – you may be surprised at what it can do.

HOW TO

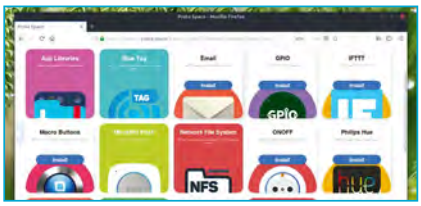
Build a Prota IoT device

1 Flash the image

Download the Prota Pi image and use the Etcher.io app to flash it to an SD card. On first boot Prota OS is in AP mode and presents as an access point.

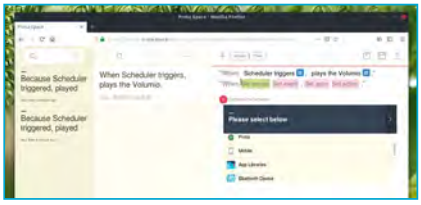
2 Configure Prota

To configure the device, connect to Prota's AP and then either head to **http://prota.space** in a browser or use the Prota Space app for Android and iOS. Follow the first-time setup wizard.



3 Install apps

Click the App Libraries link in the Dashboard to view and install any or all apps. It's these apps that IoT-ise any device connected to the Pi and some can even help automate routine tasks.



4 Create stories

One of the apps that's already installed is Stories. You can use it write rules to trigger event-based actions, such as turn on lights when you get home.

Advanced deployments

Highly specialised Linux distributions reared for very particular tasks

Linux's malleable nature is often exploited to create barebones distributions that perform singular tasks. This might be running routers (OpenWRT), software firewalls (IPFire) or gateway servers (ClearOS).

Quick servers

One such interesting distribution is TheSSS (Smallest Server Suite, **http://thesss.4mlinux.com**), which helps you deploy servers with a single keystroke. The distribution includes popular open source packages over a minimal Linux base to roll out fully fledged web, file, proxy servers and more on your local network. There are several use cases for such a quick fire server. For instance, you can deploy a FTP server to quickly share files between several users on an intranet. Or, set up a temporary proxy server to see if it helps take some load off the network. Broadcast meeting minutes or other information via the included web server. By weeding out the deployment complexities from these essential services, TheSSS opens the doors to lots of possibilities.

TheSSS runs as a Live system by default, but it can also be installed on a hard disk. Once the distribution is up and running you can enable the servers with a simple command, such as `httpd start`, to fire up the Apache web server. The fact that the distribution enables you to run the servers from memory without installation makes TheSSS very useful. It's the perfect backup server for when the main server running on a network goes down. TheSSS runs on both 32-bit and 64-bit hardware and the ISO weighs in just over 100MB. It's based on the 4M Linux distribution which in itself is a miniscule distribution that packs in a fully functional LAMP server along with the usual desktop productivity tools.

A storage pool

A simple backup server is sometimes not enough. If you seriously want to protect your data you need to use a Network Attached Storage device. While off-the-shelf NAS devices cost several hundred pounds, the Debian-based Open Media Vault (OMV, www.openmediavault.org) enables you to cobble together a NAS server of your own for free. The distro offers the power of commercial solutions in a package that's easy to set up and manage. OMV is a lightweight distro and doesn't really need a very powerful host; in fact, the project makes an image of the distribution for the Raspberry Pi. Using the Pi as an always-on NAS box sounds like a wonderful use of the silent little device.

Before installing the distribution, plug in all your storage devices to the server. If you're using the Pi you can use self-powered USB disks. Once OMV is up and running, you can configure and manage the server using its intuitive browser-based admin interface. ■

EXPERT TIP

RAID levels

OMV supports multiple RAID levels and each requires a different number of disks. For example, the default RAID level 5 requires a minimum of three disks, while RAID 1 only needs two.

Above The servers in TheSSS can be configured via a web-based interface. In addition to all these servers, the 4M Linux distribution includes several useful maintenance tools

QUICK GUIDE

Deploy OMV

Setting up OMV is fairly straightforward. You'll have to run through a quick installation if you are deploying OMV on a PC, but you can immediately jump to the setup after transferring the image to the Raspberry Pi. Bring up OMV's web-based interface and run through the setup process. Next head to the Storage section to make OMV aware of the connected disks. You can then tie them into a software RAID or format the disks if you plan to use them individually. Before you can store data on the NAS device, you have to create one or more users. The next step is to define a shared folder on the storage disks/RAID you created earlier. Finally, enable a network service that users on the network will use to access the shared folders on the NAS.

GET THE WORLD'S BEST LINUX MAGAZINE

DELIVERED DIRECT TO YOUR DOOR

Order online at **www.myfavouritemagazines.com**

or find us in your nearest supermarket, newsagent or bookstore!

SUBSCRIBE!

Sign up today and get your **FREE** Eclipse Charger

FREE GIFT WORTH £69.99

Don't miss out, subscribe now!

The Eclipse Charger, with its stylish looks and integrated cable tidy, will revolutionise how you top up your portable devices. Its Smart IC Technology auto-detects your device's input to charge at maximum speed, and you can charge up to three devices at the same time!

IT'S EASY TO SUBSCRIBE!

Click: **myfavouritemagazines.co.uk/LIN/ECLIPSELUD**

Call: **0344 848 2852**

Overseas subs on **p63**

» PLUS: Exclusive access[1] to the *Linux Format* subs area!

1,000s of DRM-free PDF back issues and articles! Get instant access back to issue 66 (May 2005) with tutorials, interviews, features and reviews. At **linuxformat.com**

1) Only available to MyFavouriteMagazines.co.uk subscribers.

» CHOOSE YOUR PACKAGE!

PRINT EDITION

» Every issue comes with a 4GB DVD packed full of the hottest distros. Plus your free **Eclipse charger**

Only £36

Every 6 months by Direct Debit

DIGITAL EDITION

» The cheapest way to get *Linux Format*. Instant access on your iPad, iPhone and Android device.

Only £11.25

Every 3 months by Direct Debit

PRINT + DIGITAL

» Get both the print & digital editions for one low price. Plus your free **Eclipse charger**

Only £43

Every 6 months by Direct Debit

*Terms and conditions: This offer is only available for new UK subscribers. Gift is subject to availability. Please allow up to 60 days for the delivery of your gift. In the event of stocks being exhausted we reserve the right to replace with items of similar value. Prices and savings quoted are compared to buying full-priced print issues. You will receive 13 issues in a year. You can write to us or call us to cancel your subscription within 14 days of purchase. Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) or are included as part of any inclusive or free minutes allowances (if offered by your phone tariff). For full terms and conditions please visit **www.bit.ly/magterms**. Offer ends 31 October 2018.

CODING A MURDERER

Discover how algorithms put serial killers at the centre of the map – **Tamsin Oxford** investigates the Murder Accountability Project

KEY INFO

Thomas Hargrove is a journalist and the mind behind MAP, the Murder Accountability Project

One journalist, a story on prostitution, a passion for open source software, and an old yet powerful database led to the creation of the Murder Accountability Project (MAP, **www.murderdata.org**). The MAP involves finding serial killers, preventing murders and connecting the statistical dots. It's also a project that has a fascinating back story.

Thomas Hargrove – the journalist – had purchased a Uniform Crime Report from the University of Missouri while doing research on a story about prostitution in 2004. The university threw in the Supplemental Homicide Report at no extra cost, and this free, data-heavy document changed the course of Thomas's life.

"The document contained row after row of information about individual murders that covered everything from the month, the year the murder happened, and the jurisdiction," says Thomas. "The file also contained data around the age, sex, method of killing, race and the police theory around the killing, plus the offender's details if the information was available.

"The moment I saw this file I was asking myself one very important question: would it be possible to use this data to teach a computer to detect serial murders?," he continues. "Could I use open source tools to build a platform that enables people to access this data and understand it in ways that allow for these murders to be solved more effectively?"

Today, MAP is a viable open source, non-profit organisation with an algorithm that is 'capable of detecting serial killers who target multiple victims using similar methods of killing within a specific geographic region'. The platform takes a depressingly endless list of violent deaths and transforms it into a visual tool that highlights patterns and trends which may have been previously undetected by the FBI or the police. It also neatly helps to solve one of the biggest challenges facing police detectives – 'linkage blindness' – where they don't recognise the link between one case and another, or that there may be a common offender involved. It's understandable: different people working different cases reported in different ways would make it hard to pick up similarities. It's something MAP can support detectives in overcoming, because it connects the data dots to create visual patterns and highlight similarities.

The project has undergone several iterations since Thomas first held the data in his hands back in 2004. It wasn't easy finding the algorithm and teaching the code to translate the data in meaningful ways. Today, MAP has evolved thanks to inventive code, determined development and a passion for solving the unsolved...

Once Thomas discovered the data in 2004 he spent the next six years trying to persuade his editor to allow him to test his theory. He knew that the information that the university had given him could be used to refine and resolve murders, he just needed the time and the budget. After years of pushing the right buttons he finally got the chance to use the data while working on a project known as Murder Mysteries in 2010.

The project won awards and raised just the right amount of awareness to allow for Thomas to take his work to the next level. The University of Missouri offered him the support of a talented Masters candidate, Liz Lucas, and so started the hunt for the perfect algorithm.

"We found hundreds of things that *didn't* work," says Hargrove. "We tested to see if an elevated rate of murders would indicate the presence of a serial killer, and it failed. We tested to see if there was an elevated rate of female murders, and this didn't deliver results. We tested to see if there was an elevated rate of murder for a particular type, such as strangulation, and this was also a no."

> "The platform takes a depressingly endless list of violent deaths and transforms it into a visual tool that highlights patterns and trends"

The team tried a lot of variations, but ultimately it was a serial killer who taught them what would work: the Green River Killer, Gary Ridgeway. He had left a series of bodies behind him as one of the most prolific killers in the United States. He strangled 49 women before he was caught and confessed to the crimes.

"We knew there had been a serial killer in Seattle and we wanted to know if we could craft a computer program that would alert us to this pattern in Seattle at that time," says Hargrove. "We tried all these different combinations to see if Seattle would appear and nothing worked. Then we tried the rates of unsolved murders, which also didn't really impact on the results. Finally, what worked was cluster analysis."

It was a blend of the victim's gender, location and method of murder that finally worked. Originally, the data used age, but this metric has been subsequently discarded because it had a minimal impact on results. The murders were then allocated a murder group number based on these four categories and the system used this structure to create around 100,000 groups.

"Then we told the computer to calculate how often, in each of these groups, the murders were solved, and to alert us to any large groups of murder clusters that had low rates of resolution," Hargrove says. "It worked like a dream. We had finally hit upon the algorithm. We then applied it to the area where the Green River Killer had been active and there it was – a huge bubble over Seattle. Another serial killer came right after Ridgeway and the algorithm detected him easily."

The algorithm was now capable of finding dozens of clusters all over the country, a plethora of deadly bubbles that indicated an excess of murder, which may or may not indicate that a serial killer is at work. Some are known, some are not.

The tech behind the algorithm

"We have used a variety of different solutions to create MAP," explains Hargrove. "We have proprietary software called Tableau (**www.tableau.com**) that enables us to visually display the data on the website, but as we're

SPOTLIGHT

Better late than never

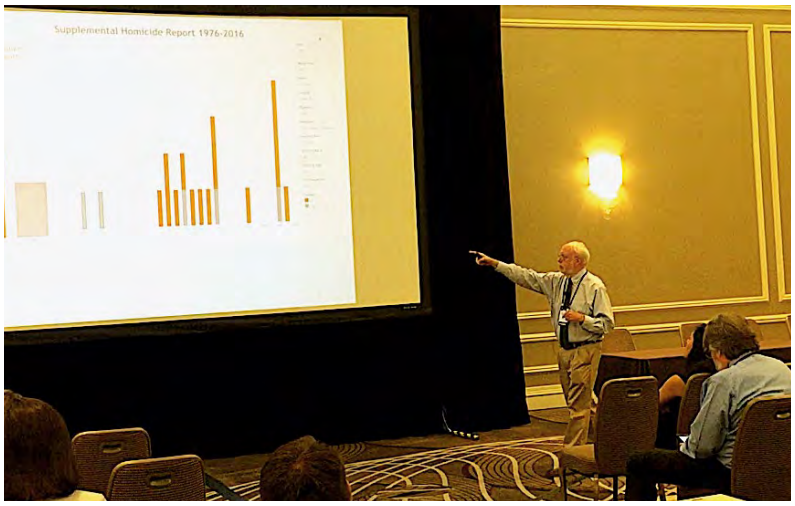
In 1974, Arlis Perry was murdered at Stanford University. Her murder had never been solved – until now. She may have been one of the victims of a serial killer only discovered years after her death, and hours before his.

In a July 2018 interview, the Santa Clara County Sheriff revealed that it was reopening this cold case after work done with the NBC Bay Area and the Murder Accountability Project found evidence that Arlis may have been the victim of an alleged serial killer, a man we won't name as proof is still being gathered and a case still being made, although the data is compelling.

"DNA testing found evidence that [the alleged serial killer] was a suspect in the 1974 murder, but when the police went to knock on [man]'s door, he committed suicide rather than be questioned," says Thomas Hargrove. "This sparked interest. Was there a chance he could have killed anybody else? Why would he have killed himself rather than face the police? To answer these questions, we went into our data systems and selected the Santa Clara jurisdiction in the time frame required. The algorithm identified all the woman who had died in similar ways."

The data showed that in the 1970s and 1980s a large number of unsolved murders appeared and then stopped abruptly after 1991. At this exact time, [the suspect] had been arrested for stealing books from Stanford, where he was a security guard. After he was released he moved to Florida – and thus the pattern ended with his departure.

"There's a reason to believe that the reason why these murders suddenly stopped was because he had moved," adds Hargrove. "Now the pattern and [the suspect] are undergoing further investigation to uncover whether or not a serial killer has indeed been found."



Above Big graph, bigger problems: murder is on the rise is the US

committed to ensuring all the data and tools are open source and accessible, we also make the raw data available to anyone to use on any platform they wish."

Visitors keen to turn their eye to unlocking patterns and potentially saving lives can use a free copy of Tableau, download the workbook created by MAP and manipulate the data. It can't be saved onto a local hard drive or server, but it does enable anyone to work with the technology and the data using the same system as MAP. The team of the non-profit organisation, now extended to include a vice chairman, treasurer and board of directors, uses Tableau because it provides them with the tech they need to display large sets of data on the website.

"We have ensured that the base data and everything that we have built onto the platform is not proprietary," says Hargrove. "We are dedicated to making homicide data more readily available to the world, which will hopefully encourage the public to review murder occurrence data and easily access the information. Our target audience is, of course, homicide detectives and we wanted a system that would allow them to call up records on their own and share them with other departments."

Open source

MAP is designed to be as open a platform as possible, to ensure it can be accessed across jurisdictions, US counties and varying technical ability. The team isn't interested in secrets or mystery or obfuscation – they use the data that the FBI has openly provided them and use it to enhance and expand their capabilities and the efficiency of the algorithm.

"The report provided by the FBI accounts for all the homicides in each jurisdiction, along with case-level detail that includes around 30 variables for each individual," says Hargrove. "We have also combined this with the data gathered by regional police departments. Many of these don't collaborate with the FBI, so we have added their information to the data that the FBI has given us to create an incredibly detailed database. We currently have case-level details on 752,000 murders from 1976 to the present. You won't find that level of insight anywhere else."

SPOTLIGHT
Getting away with murder

Above Thomas Hargrove of the Murder Accountability Project discussing the impact of data on murder resolution

Why is it important that the murder resolution rate is the lowest in history? Why does it matter that murders aren't solved? The Murder Accountability Project has the answer. The non-profit organisation occasionally does analysis of the data it owns and recent research has examined the consequences of declining clearance rates.

"We looked at every major police department that had more than 10 murders a year over five years, and what happened when you compared the five-year period ten years ago to the most recent five-year period," says Thomas Hargrove. "It found that jurisdictions that do a below-average performance solving murders suffer homicide rates nearly double of those with above-average clearance rates."

This statistic is even more of a concern in light of the fact that the FBI has estimated that homicide clearance rates dropped to 59 per cent in 2016, which is the lowest level ever recorded in the US.

"There's a broad inverse relationship between homicide clearance rates and homicide occurrence rates," concludes Thomas. "Simply put, those Americans that live in cities which don't catch the killers are twice as likely to die in a criminal assault." Food for thought...

This data is also freely accessible to the public and can be downloaded as UCR (Uniform Crime Reporting) reports. These reports are assembled using the IBM program SPSS Statistics, which is capable of managing sophisticated operations and delving into the database to impressive depths. Because this is a proprietary system, MAP has opted to use PSPP (**www.gnu.org/software/pspp**) as the open source alternative. This GNU project enables access across multiple platforms that underpins the MAP philosophy: everyone should be able to play a part in detecting serial murderers.

"We advise people to rather use PSPP because it can be downloaded at no extra cost – many people have done just that as they want to work with the datasets we have created," says Hargrove. "Making the entire project open

Thomas underscoring the value of murder accountability at the Investigative Reporters and Editors (IRE) annual conference

source is incredibly important to us as there's still so much more work to be done. The murder resolution rate has been declining and we believe that by encouraging people to come to our site and download the workbooks, raw data and data dictionaries they can call us with questions and suggestions. This is the entire principle that drives the open source community and can help us save lives."

> "Visitors keen to turn their eye to unlocking patterns can use a free copy of Tableau"

The data collected by MAP from the FBI is decidedly user-unfriendly. Written in Cobol, it comes with some oddities and complexities that impact on its accessibility, but this was the language that was common when the FBI started assembling the data.

To resolve the challenges of shifting the data from complex to accessible, MAP has written sophisticated syntax commands that take these wads of data and assembles them into the PSPP files. This can be further split out into CSV files which, Hargrove admits, may not be as useful as PSPP but are a *lingua franca* that every statistical package can open.

"These syntax commands are also available to anyone who wants them," says Hargrove. "They have made a laborious process far easier. They also enable us to expand the searches more widely – from only getting insights from a specific jurisdiction to insights across borders, because criminals don't respect geopolitical boundaries. This entire process is made very simple with the PSPP platform."

The records are assembled and put into multiple formats so they can be downloaded on demand. This is the data that is then loaded into Tableau to showcase all the information in a visual format – age, gender, method, location, type of murder and more – and allows for quick cross-tabulation. For detectives who work with limited time and plenty of information, Tableau helps them to test a theory or determine if their type of crime has occurred in different jurisdictions.

"Over the past year we have added further refinement to the system, where we now visually show murder clusters identified by the algorithm we've developed," adds Hargrove. "There's a kind of magic to carefully counting things and uncovering answers. Our algorithm counts through the three-quarters of a million records looking for clusters that have an elevated probability of being serial murders and puts them on this visual map that reveals serial murder potential."

So the system works – and now the team is overwhelmed with cases. Cases that need to be solved because, as Hargrove concludes, "Budgets, limited resources, increased death tolls – all these factors are influencing the ability of the authorities to solve crimes. What makes this more of a concern is that if murders go unsolved it inspires even more murders. Whenever a killer gets away with a crime, he becomes a living testament to a system that isn't working. Murder begets murder, and clearance reduces murder."

Sources

- SPSS Statistics
  www.ibm.com/uk-en/marketplace/spss-statistics
- Supplemental Homicide Report
  https://ucr.fbi.gov/nibrs/addendum-for-submitting-cargo-theft-data/shr
- University of Missouri
  https://missouri.edu
- Uniform Crime Reports
  https://ucr.fbi.gov

John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on and has extensive programming experience.

Resources

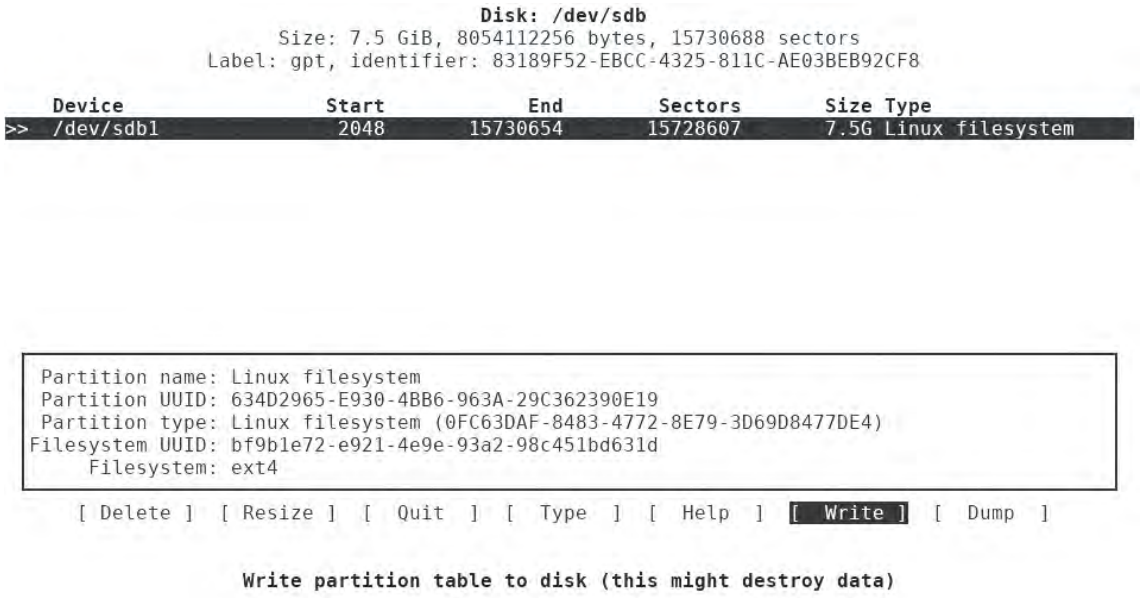
cfdisk
Probably already on your system; otherwise, install util-linux from https://mirrors.edge.kernel.org/pub/linux/utils/util-linux/v2.32

Arch Linux ISO
www.archlinux.org/download

fuseiso
https://ubiz.ru/dm/fuseiso-20070708.tar.bz2

cfdisk: Disk partitioning and Linux installations

Learn how to use cfdisk to partition disks, and how to install Linux on those new partitions


Disk: /dev/sdb
Size: 7.5 GiB, 8054112256 bytes, 15730688 sectors
Label: gpt, identifier: 83189F52-EBCC-4325-811C-AE03BEB92CF8
Device Start End Sectors Size Type
>> /dev/sdb1 2048 15730654 15728607 7.5G Linux filesystem
Partition name: Linux filesystem
Partition UUID: 634D2965-E930-4BB6-963A-29C362390E19
Partition type: Linux filesystem (0FC63DAF-8483-4772-8E79-3D69D8477DE4)
Filesystem UUID: bf9b1e72-e921-4e9e-93a2-98c451bd631d
Filesystem: ext4
[Delete] [Resize] [Quit] [Type] [Help] [Write] [Dump]
Write partition table to disk (this might destroy data)

Above cfdisk makes it easy to partition disks through its user-friendly interface. Admittedly, this might not look all that friendly with so many numbers floating around, but it's a lot easier than using the command line!

Any file storage system, whether it's a CD-ROM, a hard drive or a USB stick, is essentially a device for storing a long chain of 1s and 0s. When we interact with it, however, we don't see this; instead, we see files, directories, symbolic links and so on. The software that makes the translation for us is called a file system.

There's one step in between, though, and it's often overlooked. A particular file system need not take up the whole of the disk it lives on. Instead, the disk may be split up into individual sections, called partitions, each of which can have its own file system – or none at all. The disk keeps track of its partitions through its partition table, which tells it how many partitions there are, and where each one begins and ends.

There are a number of disk partitioning tools, but the one we shall be focusing on is one of the easiest to use, `fdisk`. In fact, we shall mainly be using its fancier cousin, `cfdisk`, which makes things easier still through its ncurses interface.

We'll be demonstrating some of the power of partitioning by installing a complete Linux system to a USB stick, while saving some space 'on the side' for file storage. The stick needs to be at least 8GB in capacity – and obviously, first make sure there's no data on it you want to keep, because we'll be removing everything.

Before partitioning your USB stick, you should find out what your computer identifies it as. Insert the stick into your computer's port (if it's already in, eject it and put it in again). Once it's in, run `dmesg` in a terminal and look for output near the end similar to the following:

```
scsi host4: usb-storage 2-2:1.0
scsi 4:0:0:0: Direct-Access Generic Flash Disk
sd 4:0:0:0: [sdb] 512-byte logical blocks
sd 4:0:0:0: [sdb] Write Protect is off
```

The thing we're looking for is the name `sdb`. Depending on your setup, you might see something different, such as `sdX` or `hdp` – but normally on a Linux system, it'll be `sd` followed by a single letter.

Let's suppose that your USB stick is available at `sdb`. You can verify this by running the `lsblk` command, which gives an overview of the storage on your system. You might see something like the output in **Figure 1**. Here, `sda` is our computer's hard disk, which is already split

into partitions sda1 and sda2 (there may be more on your system). Meanwhile, our USB stick, sdb, has a single partition, sdb1.

Use partitioning

Once you've identified the device name for your USB stick, you can fire up cfdisk:

```
# cfdisk /dev/sdb
```

If your USB stick is not at sdb, use the other value here instead. You should see a screen like the one in the main image (see facing page).

Using the arrow keys, you can navigate between the partitions currently on the stick. Each row of the table represents a contiguous range of memory addresses that is either a partition or an area of empty space. Most likely, you'll start off with a single partition and empty space, as in **Figure 1**. Use the left/right arrow keys to select the [Delete] option and press Enter to delete this partition. If there are any other partitions, use up/down to select them and then delete them as well, so that the entire USB stick is free space.

Let's start partitioning the drive. Our first partition will be 2GB in size. Use the left/right keys to select the [New] option at the bottom and press Enter. At the prompt, enter 2G, and press Enter again to create the partition.

This should have been fairly easy, so let's make a new partition. Press the down arrow to move the cursor at the top into the Free space row: we will create the next partition in here. Select [New] as before, and this time type in 250M rather than 2G, before pressing Enter to create a new partition of size 250MB.

This partition is going to play a special role: it's going to be the boot partition for our USB stick. Any bootable medium, whether it's a Live CD, your laptop's internal hard drive or a live USB stick, has to have a special partition called the boot partition that contains various files necessary for booting up the system. The boot partition may contain other files as well – indeed, various older systems are entirely contained inside one bootable partition – but it is a bit more robust to only use the boot partition for booting purposes.

For now, all we need to do is to make this partition bootable. Keeping the cursor at the top focused on your new partition (which might be named /dev/sdb2), use the left/right arrows to select the [Bootable] option, and press Enter to set the bootable flag. You should see a small star * appear next to /dev/sdb2 at the top, in the Boot column.

Now let's create another 2GB partition after the one we've just made. Press the down key to move into the Free space, and then select [New] at the bottom again, typing in 2G and pressing Enter as before. At the top of the display, we might see something like the following:

```
   Device     Boot Start       End   Size
 > /dev/sdb1        2048   4196351     2G
   /dev/sdb2  *   4196352   4708351   250M
   /dev/sdb3      4708352   8902655     2G
   Free space     8902656  15730687   3.3G
```

Having the free space at the end means that we can come back and use the [Resize] option to grow the sdb3 partition if we want to later on. Growing the sdb1 and sdb2 partitions will be much harder once all the partitions have data written to them: we will need to recreate the partition table in full in cfdisk, and then use another tool (such as dd) to shift the data into the new positions. This is both time-consuming and potentially dangerous to the data if any errors occur.

The very last step is to write this partition table to the USB stick. At the moment, nothing we have done has actually altered the state of the disk at all, which is a very good thing: if you make any mistakes, you can correct them (or select [Quit] to start again completely) and be sure that you won't have messed anything up. This is particularly important if you are repartitioning a disk that already contains data, since writing the partition table incorrectly could destroy some of that data.

> "There are a number of disk partitioning tools, but the one we shall be focusing on is cfdisk"

Using the left/right keys, select [Write] at the bottom, and press Enter. cfdisk will check one last time that this is what we actually want to do:

```
  Are you sure you want to write the
partition table to disk?
  Type "yes" or "no", or press ESC to leave
this dialog.
```

Type yes and then press Enter to write the partition table. Lastly, navigate to [Quit] and press Enter to close cfdisk. We can check what we've done by running lsblk again. After these changes, the sdb portion of the output should look something like this:

```
sdb         8:16   1   7.5G  0 disk
|—sdb1      8:17   1   1.9G  0 part
|—sdb2      8:18   1   238M  0 part
|__sdb3     8:19   1     2G  0 part
```

Figure 1

```
NAME      MAJ:MIN  RM   SIZE   RO  TYPE  MOUNTPOINT
sda       8:0      0    140.0G 0   disk
|—sda1    8:2      0    260M   0   part  /boot
|__sda2   8:8      0    139.5G 0   part  /
sdb       8:16     1    7.5G   0   disk
|__sdb    8:17     1    7.5G   0   part
```

Above We can use the lsblk command for a useful high-level overview of the partitions on the devices on our system

Alternatives to cfdisk

There are many different disk partitioning tools. You might want to give cfdisk's predecessor, fdisk, a try: you'll have to type commands, such as d for delete, rather than selecting buttons on an interface. A more powerful, but slightly more difficult tool is parted; if you want to use this, you might consider installing GParted, which gives you a full-blown graphical interface.

Figure 2

```
# mount --bind squashfs-root squashfs-root
# mount -t proc none squashfs-root/proc
# mount -t sysfs none squashfs-root/sys
# mount -o bind /dev squashfs-root/dev
# mount -o bind /dev/pts squashfs-root/dev/pts
# cp -L /etc/resolv.conf squashfs-root/etc
```

Above In order to chroot into the Linux system on disk, we need to connect the 'magic directories' to their counterparts on the base system

At the moment, our partitions aren't doing anything. All we've really done is to alter something called the partition table, which tells the disk where each partition begins and end. If we want to start using the partitions to store files, we need to format them as file systems and then 'mount' them to the operating system.

The standard file system for Linux is called Ext4, while boot partitions are often formatted using the FAT32 system (because FAT32 tends to be supported by a wide number of hardware devices and non-Linux-based operating systems). We can create these file systems using the `mkfs.ext4` and `mkfs.fat` commands:

```
# mkfs.ext4 /dev/sdb1
# mkfs.fat -F32 /dev/sdb2
```

> "A nice feature of Linux is that we can run other Linux installations within the main one"

```
# mkfs.ext4 /dev/sdb3
```

You'll need to wait a few seconds, but soon the files system will be present. You might want to try mounting one of these partitions – that is, make it available to be used by the operating system – so that you can test out the file system. You can do this using the `mount` command, like so:

```
# mount /dev/sdb1 /mnt
$ cd /mnt
```

Now our partition sdb1 is mounted as the `/mnt` directory, and we can explore it, add new files and so on:

```
# touch a.file
$ ls
a.file  lost+found
```

Don't go overboard adding files, though, since we will be overwriting all the data in this partition. Once you've finished, navigate out of the `/mnt` directory and unmount the partition using `umount`. There are a couple of ways to do this – use

```
# umount /dev/sdb1
```

or, if you prefer:

```
# umount /mnt
```

The results are exactly the same.

Installing Linux to a partition

We'll demonstrate some of the power of partitioning by installing a very basic Arch Linux distribution to the sdb1 partition. Arch Linux is a good choice for this, since it can easily be installed from the command line. First, you need to download the Arch ISO image, which you can find at **https://www.archlinux.org/download**. Move this file into a new empty directory where you can do some work. For example:

```
$ mkdir arch
$ cp ~/Downloads/archlinux-2018.08.01-x86_64.iso arch
$ cd arch
```

Inside this directory, create a directory called `arch`. We'll mount the ISO image to this directory using the `fuseiso` tool like so:

```
$ fuseiso archlinux-2018.08.01-x86_64.iso arch
```

The ISO image is now be mounted in the `arch` directory. It essentially contains a minimal Arch Linux distribution. A nice feature of Linux is that it enables us to run other copies of Linux within the main copy, without having to boot up from an installation medium, and the next few steps will show how we do this. It is worth mentioning, however, that it might be easier to burn the ISO image to a disc and boot from that, before skipping to the installation instructions in the next section.

Once we have the ISO image mounted at `arch`, we can copy out the file we need, `airootfs.sfs`, before unmounting the ISO:

```
$ cp arch/arch/x86_64/airootfs.sfs .
$ fusermount -u arch
```

The `airootfs.sfs` file is compressed using the SquashFS file system. The first step is to unsquash (decompress) it:

```
# unsquashfs airootfs.sfs
```

This creates a new directory, `squashfs-root`, containing the bare-bones Arch Linux system. The next step is to connect the 'magic' file systems `/dev`, `/sys` and `/proc` to their counterparts on our main system. These directories don't contain files in the traditional sense; they're meant to be linked to hardware objects and system resources.

For example, on our main system, `/dev/sdb` is the location of our USB stick. If we want to be able to refer to this device as `/dev/sdb` on our new system, we'll need to connect `/dev` on the main system to `/dev` on the new one. We do all this using the `mount` command, as in **Figure 2**. Lastly, we can use the `chroot` command to log on to the

Arch installation that is living inside the **squashfs-root** directory:

```
# chroot squashfs-root bash
```

We find ourselves dropped at a terminal: to all intents and purposes, we are now running on the new system.

Installing Linux to the USB stick

By this point, you should be logged into a bare-bones Arch Linux system – either if you've followed the instructions above, or if you've booted off the ISO image directly. At the moment, there is next to no software installed on this system, so you need to use Arch's **pacman** package manager to install some basic programs:

```
# pacman-key --init
# pacman-key --populate archlinux
# pacman -Syyu
# pacman -S base base-devel
```

Some of the commands might produce a bit of output; make sure you run them all, though.

One problem we need to fix is that the partition table is currently in MBR mode, whereas most bootloaders require a GPT partition table. MBR is a much older format, and is less capable than GPT. To make the switch, use the **sgdisk** command, which should already be installed on the Arch system (if not, run **pacman -S gptfdisk**):

```
# sgdisk -g /dev/sdb
```

We now need to mount the partitions we created earlier inside the new system, which we do using the **mount** command as before. You can mount the root directory **/** to the main partition and the **/boot** directory to your boot partition, like this:

```
# mount /dev/sdb1 /mnt
# mount /dev/sdb2 /mnt/boot
```

You should now be able to run the following commands to install Arch Linux on the USB stick.

```
# mkdir -p /run/shm
# pacstrap /mnt base
```

This step should take a bit of time. When it has finished, Arch Linux should be installed on the USB stick. We still need to install some additional software on our new installation in order to make it bootable. First, do another **chroot** to take you to the new installation of Arch that you created on the USB stick:

```
# genfstab -U /mnt >> /mnt/etc/fstab
# arch-chroot /mnt
```

You should be dropped at a new terminal, and are now effectively running Linux off the USB stick. Sometimes there's a problem with the installation of the **linux** package by **pacstrap**, so run **pacman -S linux** to reinstall it. Now we can install the bootloader using the following command.

Figure 3

```
# cat > /boot/loader/entries/arch.conf
Name Arch Linux
linux /vmlinuz-linux
initrd /initramfs-linux.img
options root=/dev/sdb1 rw
<Ctrl-D>
```

Left If you want to install multiple Linux distributions to the same medium, you should create a configuration file like this for each of them

```
# bootctl --path=/boot install
```

The last step is to create an boot entry for the new installation, which we do by writing appropriate contents to a new file, **/boot/loader/entries/arch.conf**, as in **Figure 3**.

The installation should now be set up on the USB stick. Press **Ctrl+D** to leave this **chroot** environment, and then run **umount /mnt/boot /mnt** to close down the mount points. Then press **Ctrl+D** to leave the original **chroot** environment, and run **sudo umount -l squashfs squashfs/*** to remove the mount points we created earlier. Lastly, shut down your computer and boot it up from the USB stick. You should be able to boot up into a very basic version of Arch Linux.

> "It's up to you to create more partitions and install more Linux systems to the USB stick if you want to"

It's up to you to create more partitions and install more Linux systems to the USB stick if you want to take things further. You should use a different partition for each installation, though you should use the same boot partition for all of them.

Installing Linux through a graphical tool

Many Linux distributions – for example, Antergos and the various flavours of Ubuntu – provide you with a graphical tool for installing Linux to new media. For this, we recommend booting from a Live CD and then running the tool. At some point, you should have the option to choose the partitions to which to install Linux. For Antergos, for example, you can select 'Choose exactly where Antergos will be installed', and you then have the option to choose the correct main and boot partitions. It's always useful to know how to do this manually, though.

Shashank Sharma

Shashank is a trial lawyer in New Delhi, and has been writing exclusively about open source software for the past 14 years.

Resources

S-TUI
www.github.com/amanusk/s-tui

S-TUI: Stress-test your CPU from the terminal

The nifty little S-TUI is an elegant command-line application to monitor your CPU and run a stress test

Above Enable the 'Smooth Graph' option on the sidebar for a better graphical representation of the collected data

The Linux command-line offers a vast array of useful applications for a host of different use cases. Everything from everyday text editing, connecting with remote machines, performing backups and other administration tasks can be done from the terminal. Monitoring your system's performance is no exception. Depending on your purpose, you can choose from a variety of dedicated applications to monitor the different components such as CPU frequency, temperature, memory utilisation and so on.

With S-TUI, which is an abbreviation for Stress-Terminal UI, you can simultaneously monitor CPU temperature, frequency, power and utilisation. It presents all the information graphically and can even be used to export the data into CSV files. Better still, you can configure the tool to automatically launch scripts when the values of any of the components being monitored breaches the defined threshold values. When coupled with stress, another command-line utility, the tool can also stress-test your system.

Not many Linux distributions carry S-TUI in their software repositories, but installation is fairly straightforward. The project's GitHub page describes various installation techniques. If your system is already configured to use pip, you can install S-TUI with the `sudo pip install s-tui` command. If you decide to install it from the Git repositories instead, you'll have to manually install its dependencies – urwid and psutil. You can install these Python packages using your distribution's package management tool.

You can now begin using the tool to monitor your system's performance. If you would also like to stress-test the CPU, S-TUI supports both the stress and stress-ng utilities. Most distributions ship these in their software repositories, and you can install them using your distribution's package management tool.

Use S-TUI

Unlike many other command-line utilities, S-TUI requires no configuration and you can begin using it immediately after installation. Run the `s-tui` command to launch the tool. At the top of the sidebar on the left are the details about your CPU. Our Lenovo test machine was correctly identified as Intel Core i3-5005U CPU @ 2.00 GHz. S-TUI uses various other native tools and utilities to gather the relevant information; for instance, the same information, and far greater details, can be ascertained from running the `cat /proc/cpuinfo` command.

When you first launch S-TUI, it display all four parameters – Frequency, Utilization, Temperature and Power, and refreshes the data for each every second. You can change the refresh rate by changing the value of `Refresh[s]:1.0` on the sidebar.

Depending on your terminal application, and the colour scheme, you may not be able to clearly see the different elements on the S-TUI interface. Should this happen, switch to a basic White-on-Black colour scheme and run S-TUI. This should tell you all the colours used by the tool and you can then make changes to your profile accordingly.

You can navigate the sidebar using the H and J keys, much like you would in vim, or use the Up and Down keys on the keyboard. The default mode for the tool is to monitor your system, which is evident from the `(X)` next to `Monitor` on the sidebar. If you installed the stress utility as well, you can enable it by navigating to the **Stress** entry on the sidebar and pressing Space. You'll notice the empty brackets are replaced with `(X)` now.

Above Some information, such as Perf Lost, is only available when you run the tool as root user

You can similarly enable or disable monitoring of the different components. When you disable a component, the corresponding graph will automatically disappear from the interface.

Unlike most other command-line utilities, S-TUI can also be controlled with the mouse. Left-click on an entry to select it. You must still hit Escape to return to the main screen and use the arrow keys to scroll through the sidebar, as the scroll-wheel on your mouse won't work with S-TUI. If for some reason you want to disable the mouse, start S-TUI with `s-tui -nm`.

You can also switch to a different temperature sensor from within the tool itself, if you believe the graph is inaccurate. Navigate to the `Temp Sensors >` entry on the sidebar, and press Enter. This opens the Available Temperature Sensors dialogue, and you can then select one from the list. As before, after navigating to an entry in the list, press Spacer to select it. Remember to select Apply for the changes to take effect.

Hit Escape to return to the main screen of the application, such as from the About, Help or Temp Sensors section. While the graphical interface for the tool also has a Help section in the sidebar, it provides little more than a quick introduction. For a complete list of all the supported command options, run `s-tui help`.

Further configuration

If you want to save the result of the performance monitoring, use `s-tui -c`. This launches the utility as before, but when you now exit the tool by pressing `Q`, the collected data will be stored in the user's home directory. You can then view the `s-tui_log_<TIME>.csv` file in your preferred text editor. You can also provide a custom name for the CSV log file with `s-tui -c <path_to_CSV_file>`.

Another useful command option is `-j`, which can be used to show the current status of the machine in JSON format, like this:

```
$ s-tui -j
{
      "Top Freq": "2000 MHz",
      "Cur Freq": "883.0 MHz",
      "Perf Lost: "0 (N/A) run sudov,
      "Max Temp": "42.0 C",
      "Cur Temp": "42.0 C",
      "Utilization": "33.3 %",
      "Max Power": "5.0 W",
      "Cur Power": "4.3 W",
      "Fan": "0.0 RPM"
}
```

S-TUI supports a number of different temperature sensors and you can choose a different one by navigating to the Temp Sensors screen. This is only needed if the default configuration fails to accurately graph the CPU temperature (for example, if the readings seem significantly way off).

All the basic settings , such as the refresh rate (in seconds), and which components to graph by default are stored in the `~/.config/s-tui/s-tui.conf` file.

Above Refer to the man page and documentation on stress utility to understand its supported functions

Unfortunately, you must set the Stress options and the temperature sensors manually every time you run S-TUI, as the tool doesn't permit saving these settings in the configuration file.

Threshold scripts

In addition to the `s-tui.conf` file, the `~/.config/s-tui` directory contains a directory called `hooks-d`. Within this directory you can place custom scripts you want to execute when a certain threshold, such as CPU temperature, frequency or similar, is exceeded.

S-TUI supports tracking of four different sources and you can use any of these sources to create custom scripts. The name of the script must be `<name>source.sh`, where `<name>` is the name of the supported source. While you can use S-TUI to track various sources, for now it only supports running scripts based on CPU temperature threshold setting.

The default threshold value for the CPU temperature is 80 degrees C. To change this value, invoke S-TUI with `s-tui --t_thresh <value>`. You can then save the new threshold value into the `~/.config/s-tui/s-tui.conf` file by saving the settings from within the tool itself. When you now open the `s-tui.conf` file in a text editor, you'll find a new entry showing the newly defined temperature threshold value:

```
[GraphControll]
refresh = 1.5
utf8 = True
frequency = True
utilization = True
temperature = True
power = True
[TempControll]
threshold = 70
```

While the stress-testing feature is probably of little use to home and casual Linux users, there's no denying the benefit of graphical representation of the collected data for administrators and programmers. On the other hand, it's also a useful way of seeing what's going on with your CPU if you suspect it's being throttled in some way, or otherwise not performing as it should.

Pile on the stress

The stress utility performs a loop that calculates the square root of a random number in order to stress the CPU. You can customise options by navigating to the Stress Options screen from the S-TUI sidebar. The graphical interface enables you to define all the major settings such as the worker count for the sqrt(), sync() and malloc() functions, as well as choosing whether to dirty the memory.

Toni Castillo Girona

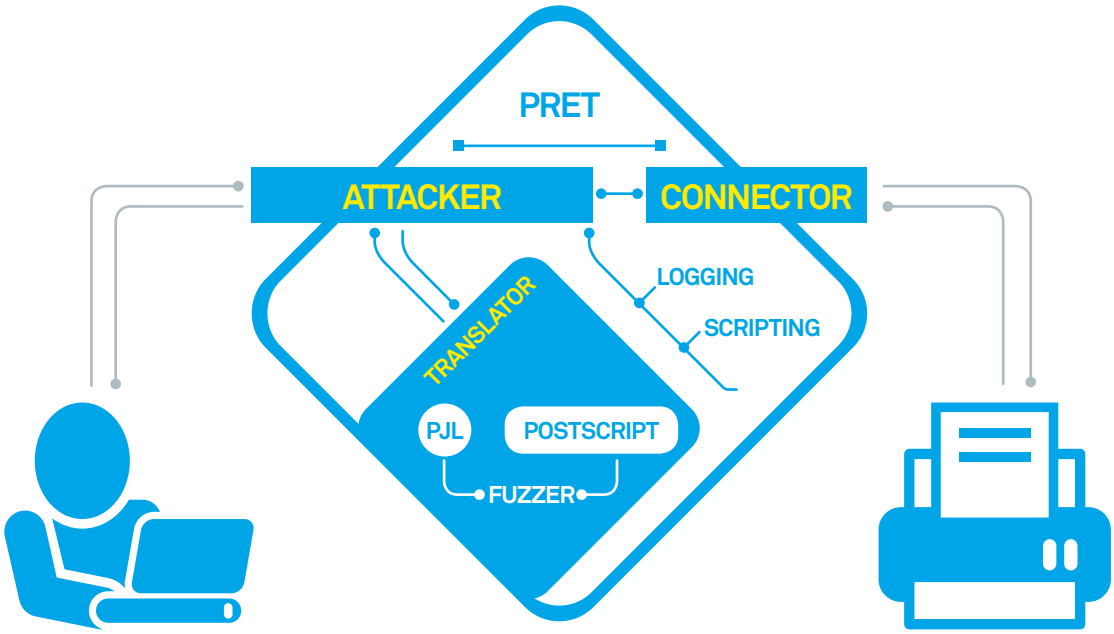
Toni has a degree in Software Engineering and a MSc in CompSec and works as an ICT research support expert in a university in Catalonia (Spain). Read his blog at **http://disbauxes.upc.es**.

Resources

- **John the Ripper Jumbo version** http://bit.ly/lud_john
- **Crowbar** http://bit.ly/lud_crowbar
- **PRET** http://bit.ly/lud_pret
- **Exploiting printers** http://bit.ly/lud_printexploit

CompSec: Remove the threats to your system

Make things a bit harder for cybercriminals by getting rid of your 'low-hanging fruit' before it's too late



Above Teach yourself a bit of the PS, PCL and PJL languages. Too daunting a task? No fret – use PRET!

Hacking into a system does not necessarily require computer-savvy people, well versed in software exploitation. More often than not it's a case of leaving bread crumbs and 'low-hanging fruit' directly exposed to the internet for all the bad guys to see (and, of course, take advantage of).

Whether it's an old unpatched system or an easy-to-guess password, that's all they need in order to get their first foothold into the system and start their lateral movement and system pivoting. The media is full of such cases, like the hacked GitHub account of Gentoo Linux (see **http://bit.ly/lud_gentoo**). Having guessable passwords, and not using public key authentication or 2FA authentication mechanisms, is a terrible mistake that typically ends up in a data breach. Running old software (sometimes even on modern OSes) poses a security threat too.

So this tutorial will guide you through some real cases where the bad guys were successful in hacking into some systems by spotting, and then exploiting, some very tasty low-hanging fruit. No advanced skills are needed, just some Bash or Python scripting and some tool or another. From old DES-encrypted passwords to leaked NTLM hashes on an LDAP schema, all the bad guys had to do was reach out...

Rehash old hashes

Think about all those servers and workstations you've been upgrading since the very first time you installed them. Like, say, the very first web server your company had running, on a flamboyant Debian Sarge with Apache Server 1.X. Since those old heady days a lot of things have changed, of course, and you've been upgrading that old web server to this day. But what about the hashes? Unless you force your users to change their passwords regularly (in case you are still using password authentication, that is), by upgrading a system you won't be upgrading those old hashes algorithms at all.

On some occasions I've seen DES-encrypted passwords happily coexisting with salted SHA512 hashes on modern Debian Stretch servers – no kidding. Why? Because the system manager did not think to make their

Right Whenever you set up pass-phrases for your LUKS-encrypted devices, go for the Snowden approach: it delivers

users change their passwords and, let's be frank here, because users tend not to change them at all unless forced to. Unless it's after a data breach, of course.

One thing you can do right now is to make sure all your servers and workstations do not have old DES-encrypted passwords or MD5 hashes. Log in to these computers and have a look at the `/etc/shadow` file. Current GNU/Linux systems use salted SHA512 hashes. If you spot any non-SHA512 hash, run the following command to force those particular users to change their passwords the next time they log in:

```
chage -d 0 <USER>
```

Get the file `01-hashes.txt` from the coverdisc. This is a recreation of an `/etc/shadow` file found on a real email server running Debian Stretch. This server was originally running Debian Sarge, and the system admin upgraded it following the obvious path: Lenny > Squeeze > Wheezy > Jessie > Stretch. Some users, though, hadn't changed their passwords since the very first time, not even the system admin himself! Brute-forcing DES with John The Ripper (see Resources) is more than feasible, so clone its repository and have fun cracking these passwords!

Get rid of the usual suspects

Every single GNU/Linux system ships with some well-known users, such as www-data, git, root and so on. For those servers with password-based authentication mechanisms, if users' passwords are easy to guess, it might as well be game over. We have found on a bunch of servers that some of these users, in charge of running some network services (like the Apache web server), were allowed to execute a shell. On one such occasion this was because the server was running an OwnCloud instance and they needed the www-data user to perform some administrative tasks running the occ PHP script via an SSH connection.

The www-data user had a guessable password and the SSH server was using password-based authentication. So it came as no surprise that one day the system admin noticed that the web server was having some serious trouble in servicing pages to clients. It turned out that someone had been able to gain a non-privileged shell as www-data by guessing the password, and they had started running brute-force attacks against other computers and networks.

Whenever possible, don't allow valid shells to daemon accounts and make sure to rename them. Allow access to those users who really need to use SSH by editing `/etc/ssh/sshd_config` and adding an `AllowUsers` entry, in the following way:

```
AllowUsers <user1@IPADDR> <user2@IPADDR>
```

Of course the IP address can be spoofed, but if you combine public key authentication with AllowUsers

Figure 1
Enter a passphrase to unlock the volume
The passphrase is needed to access encrypted data on SMI USB DISK (7.8 GB Drive).
Password:
Forget password immediately
Remember password until you logout
Remember forever
Cancel
Connect

> "On some occasions I've seen DES-encrypted passwords coexisting with salted SHA512 hashes"

entries, you should probably be fine. You can even increase your security by combining 2FA mechanisms, public key authentication and LUKS.

Carrying around a non-encrypted laptop is a perilous business. On one occasion, someone stole a laptop that had no encryption at all. They then bypassed the entire INIT process and got access to the user's home directory. There they found a public/private key pair in the `~./ssh/` directory. Running the following command, they got a list of valid hosts to which the user regularly connected:

```
cat /home/<USER>/.bash_history\
> |egrep --color=yes "ssh|scp|sftp"
```

Finally, they tried to connect to those hosts by running:

```
ssh -i <ID_RSA_KEY> <USER>@host
```

The user did not have his private key encrypted, so no pass-phrase was needed in order to use it, and so the bad guys were able to hack into a bunch of servers.

Now, imagine this happens to you. Do you have, at least, a strong pass-phrase protecting your private keys? Do you have encryption at all on the volume where you store your private keys? You can use John The Ripper to brute-force private keys just as easily, of course. Generate a new RSA public/private key-pair, and when asked for a pass-phrase, type something dictionary-based such as `mariobros`:

```
ssh-keygen -t rsa -f id_my_weak_rsa
```

Install John The Ripper JUMBO version on your computer and then execute the following:

Combining 2FA, pub key and LUKS

You can improve the security of your SSH servers a lot by combining 2FA services (such as Google Authenticator) with public key authentication. This will certainly render brute-force attacks useless. Make sure you store your private keys on LUKS-encrypted devices and have a strong passphrase to unlock them.

Exploring your system further

There are hundreds of tools and frameworks out there to assess the security of your computer, but none like CHIPSEC (**http://bit.ly/lud_chipsec**). This amazing framework will help you scan your computer deeper than any other solution in the search of potential vulnerabilities.

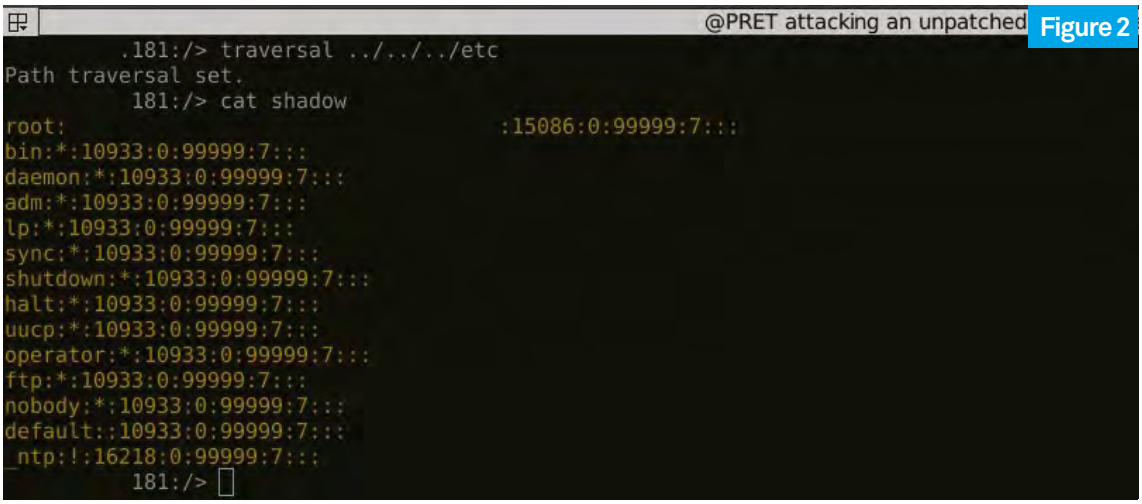


Figure 2

Above MFPs are multi-functional devices indeed: here you are about to gain a remote shell. Smashing!

```
./ssh2john.py id_my_weak_rsa > id_rsa.hash
```

From this point on, it's just a matter of running John with a good dictionary and wait for the best (or the worst):

```
./john -wordlist=/path/to/dictionary.txt \
> id_rsa.hash
mariobros          (id_rsa)
```

Feel free to crack the private key included on the coverdisc (`03-id_rsa`).

On the other hand, let's imagine you did have your entire file system encrypted with LUKS. Unfortunately, if your chosen pass-phrase is weak, you are still doomed. To demonstrate this, grab a spare USB stick, create a new VFAT partition on it and then run the following commands to create a new LUKS-encrypted volume. In this case when asked for a pass-phrase, choose a weak, dictionary-based one:

> "Printers can pose a tremendous threat because they can be exploited and used as a pivot"

```
cryptsetup luksFormat /dev/<DEVICE>
cryptsetup luksOpen /dev/<DEVICE> MYLUKS
mkfs.vfat /dev/mapper/MYLUKS -n dataencrypted
cryptsetup luksClose MYLUKS
```

Now you've got yourself a fancy VFAT LUKS-encrypted device. Better, right? Well, trying to perform password guessing on LUKS devices is admittedly slow, but you can always dump the LUKS header with all its keys to a file and work against this file to speed up the process. Bruteforce-luks (see Resources) is perfect for this purpose; first install it:

```
git clone https://github.com/glv2/bruteforce-luks.git
apt-get install dh-autoreconf libcryptsetup-dev
cd bruteforce-luks
./autogen.sh
./configure && make
```

Next, dump the LUKS header from your USB stick and run bruteforce-luks against it with a good dictionary:

```
sudo dd if=/dev/<DEVICE> of=luks-header bs=1M count=10
./bruteforce-luks -t 2 -f <DICTIONARY> luks_header
```

When the only way to access your SSH servers is by means of public key authentication, any attempt at password brute-forcing will irremediably fail. However, attackers will inevitably keep on trying. That's why it's important to install a tool like fail2ban, to ban those annoying attempts.

There's also the possibility of a private key brute-force attack by using, for example, Crowbar (see Resources). When a non-valid public key is offered, the `/var/log/auth.log` file will have entries like the following: Jul 22 08:28:42 server1 sshd[19289]: Connection closed by <IPATTACKER> [preauth]. It's easy to track these attempts and ban them with fail2ban. Just edit `/etc/fail2ban/filter.d/sshd.conf` and append the following line to `failregex`:

```
^%(__prefix_line)sConnection closed by
 <HOST> \[preauth\]$
```

Once the new rule is added, use `fail2ban-regex` to make sure it's working:

```
fail2ban-regex /var/log/auth.log \
> /etc/fail2ban/filter.d/sshd.conf
```

Do RSA weak keys ring any bells? It's something from the past (see **https://jblevins.org/log/ssh-vulnkey**) but it's still pestering our present. Old embedded SSH devices with no firmware upgrades are still affected by this issue. And guess what – some of those weak RSA2048 keys are still found in **./ssh/authorized_keys2** files! Make sure you do not have any old embedded device (router, IoT or whatever) running a vulnerable OpenSSL version and, in case you do, make sure all the vulnerable keys have been removed.

Otherwise, an attacker could download pre-calculated RSA2048 private keys and use them to gain illegal access to one of these devices. Try it; download pre-calculated RSA2048 private keys from **http://bit.ly/lud_rsakeys**. Then download and install the aforementioned Crowbar to brute-force these devices using the downloaded keys: `git clone https://github.com/galkan/crowbar.git`. Install its dependency first: `apt-get install python-paramiko` and try your luck with:

```
./crowbar.py -b sshkey -s <IP_DEVICE/32> \
  > -u <USER> -k /path/to/rsa/2048/
```

> "You won't be upgrading old password hashes just by upgrading the system"

Does that seem unlikely? We've seen it happen. Twice.

Pen-test your printers

You probably have some printers on your network. Hopefully, these printers won't be accessible from the internet but they pose a tremendous threat nonetheless because they can be exploited and used as a pivot. PRET (see Resources) is an amazing pen-testing tool for printers. It allows you to perform some pen-testing tests and exploitation of PS, PCL and PJL printers.

What sort of attacks can be possible on a printer, you may be thinking? Browse to **http://hacking-printers.net/wiki/index.php/Main_Page** to find out. Are your printers updated to their latest firmware? Are they exposed to the internet? One of our engagements took place some time ago when we spotted some odd connections from a remote IP address to an HP OfficeJet 8210 printer on an unusual TCP port. It turned out the printer had been hacked and someone had been able to spawn a netcat listener on TCP port 1210. They basically exploited the vulnerability reported at **http://bit.ly/lud_hpexploit**, by following, step-by-step, the post on HP's own website at **http://bit.ly/lud_rootprinter**. See? Even if they were not well-versed in printer hacking, they did it anyway. But of course you can even do it quicker using PRET!

Let's find out which printers you can see on your network. First, install PRET dependencies: `pip install colorama pysnmp; apt-get install imagemagick ghostscript`. Next, execute PRET to get a list of your printers: `cd PRET/; ./pret.py` (for this command to succeed you need to allow UDP packets to your computer). Pick one and re-run PRET with the language that your printer supports (ps, pcl or pjl):

```
./pret.py <IP> ps
```

Some printers may support more than one of these languages, and some vulnerabilities may be successfully exploited by using one but not the others. Bear in mind that by using PRET your printer may well be gushing out a lot of pages with odd PS/PCL/PJL codes printed on them. Trying to perform path traversal is a good starting point whenever assessing the security of your printers. If you can do it, the chances are that you may have a potential attack vector to investigate:

```
traversal ../../../bin
Path traversal set.
```

Sometimes you won't be able to do that, but maybe you will be able to use the printer as a sort of covert repository for spreading malware, illegal content storage and so on. A bunch of printers are vulnerable to this particular flaw although some vendors do not categorise it as a proper vulnerability. It's easy to test for that; using PRET, try to upload to and then download a file:

```
./pret.py <PRINTER> pcl
put <FILE>
ls
get <FILE>
```

Feel free to explore all of the possibilities of PRET and read more about printer exploitation at **http://bit.ly/lud_rootprint2**.

WHAT NEXT?

From debugging to exploiting

There are a bunch of GDB plug-ins that aid with the tedious task of debugging binaries, while providing some useful tools for exploit development and testing along the way.

1 PEDA

http://bit.ly/lud_peda

We have used PEDA a lot in the past for solving CTFs. Its pattern matching for offset calculation is a very nice feature.

2 PEDA-arm

http://bit.ly/lud_pedaarm

Now you can debug ARM binaries using PEDA, thanks to PEDA-arm. It's your chance to find any flaws on Android-based binaries!

3 pwndbg

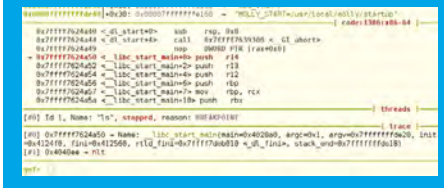
http://bit.ly/lud_pwndbg

This plug-in also helps a lot with exploit development and it even ships with Radare2 and IDA Pro integration out of the box. We don't care much about IDA, but having r2 integration is very nice.

4 GEF

https://github.com/hugsy/gef

GDB Enhanced Features is an amazing GDB plug-in that is architecture-agnostic, so it's ideal for debugging and exploiting development and testing.



Joey Bernard

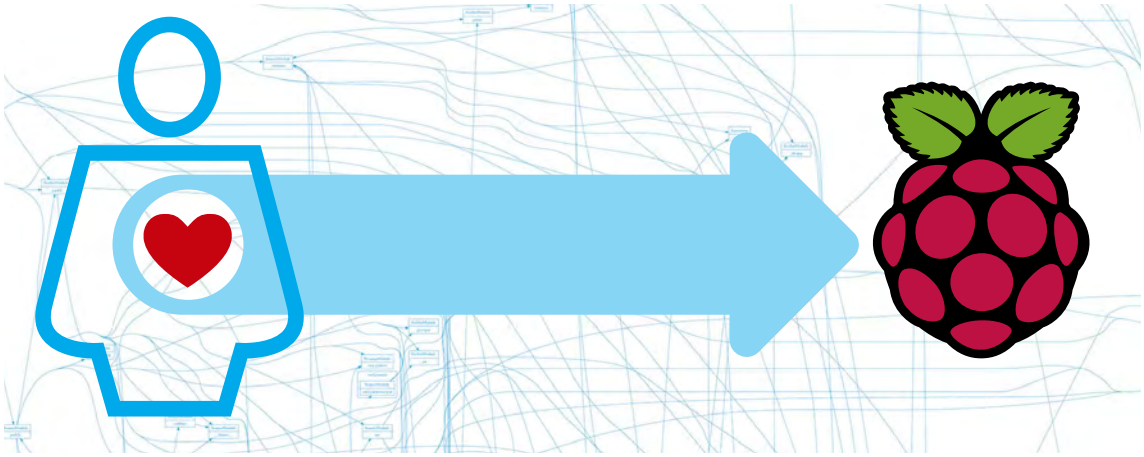
Joey is a columnist covering scientific software, as well writing Python code for the Raspberry Pi. In his day job, he helps researchers and students at the university level design and run HPC projects on supercomputing clusters.

Resources

- **PyInstaller** www.pyinstaller.org/index.html
- **py2exe** www.py2exe.org
- **cx_Freeze** https://anthony-tuininga.github.io/cx_Freeze/

PyInstaller: Learn how to share your Pi projects

This issue, we look at one of the options available to make sharing your Raspberry Pi projects easier



We've looked at several techniques in Python programming that you can add to your own project, but once you have a project put together, how do you share it with the broader audience of DIY and maker people out there? In this tutorial, we'll look at one of the options available to package your Python program, called PyInstaller. This is designed to package together everything you would need in order to run your code on a different machine. This is probably the easiest way for a new user to pick up your project and start using it.

We'll see how to install PyInstaller, how to configure it for your project, and also some possible problem areas that might come up in special cases. In addition we'll look at some other options, such as py2exe, which offer a different set of capabilities that could fit your requirements better. As in most cases, there are different tools because people have different requirements that need to be fulfilled.

Get started

So what does PyInstaller actually do to package your project up? Normally, you give it a single Python script that is the starting point for your project. PyInstaller then goes through it and collects a list of all of the other modules and scripts used by this main script. All this code is then copied into a distribution directory, along with the active Python distribution being used on the system. At this stage, you should have everything you need to run the project on a similar system.

A similar system, in this case, means the same type of hardware and the same operating system. This means that if you want to create a package for a particular system, say macOS, you will need to build the PyInstaller package on that same system – you can't make a 'universal' package that works on everything.

Once all the files are collected into one location, you can create an executable file which includes all the files needed. There are several options available to control how these files are unpacked on the end-user's system. We will look at a few of them later in this article, to see what options you might find most useful.

Install PyInstaller

Most distributions do not include PyInstaller as a module managed by the system package management system, which means you'll need to install it using `pip`. Most Linux distributions have different versions of pip for Python 2.x and Python 3.x. For example, if you wanted to install PyInstaller for Python 3, you could use the following:

```
sudo pip3 install pyinstaller
```

This installs PyInstaller to the system Python library. If you want to have the latest development version, you can hand a URL to pip, as in the following example.

```
sudo pip3 install https://github.com/pyinstaller/
pyinstaller/tarball/develop
```

Once this is done, you should everything you need to start packaging up your project. You can verify that everything installed correctly by running the following:

```
pyinstaller --version
```

With PyInstaller installed, you can use it right away with very little effort – for small projects, at least. Let's say your project all fits inside a single Python script and uses just a few of the most common modules. In this case, you can simply change directory to the one where your script resides and run the following command:

```
pyinstaller myscript.py
```

This creates a subdirectory called `myscript`, and places all of the requirements for your script there. This includes the Python engine that was used to run your script. To run your program, an executable is also created, called `myscript`, that launches your code for the end user. Sharing is as easy as zipping up the entire folder and sending it to interested parties. They then simply have to unzip the file and run the executable.

One big advantage to this method, at least for initial development, is that everything that PyInstaller thinks you need is copied into a single location. You can then verify that it did the right thing very simply by going through this destination directory. If PyInstaller did make some mistakes, you can easily figure out what these mistakes are and correct them before they go out. Once you're confident that everything is working correctly, you can use a different option to create a single big executable that contains everything needed, as in the command below:

```
pyinstaller --onefile myscript.py
```

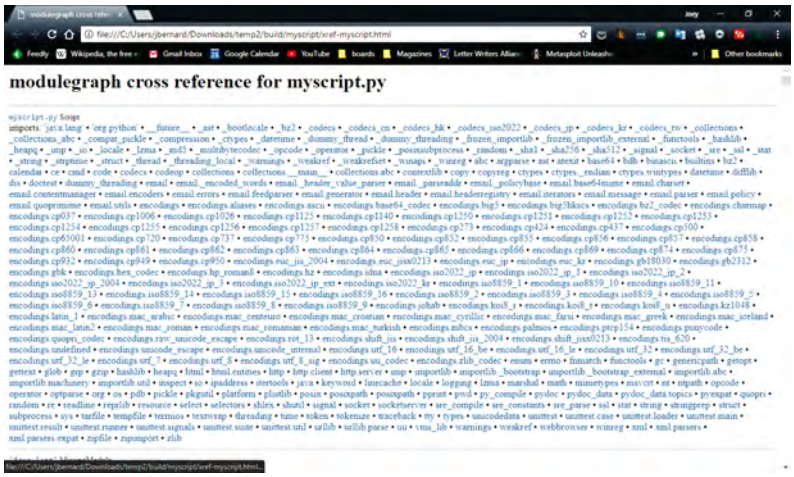
This single executable file can then be sent to your users. This is a much more comfortable experience for them, so you will likely want to use this method as the final delivery option. When your user runs this executable, it self-unpacks into a temporary directory, and then runs the executable, just as in the folder method above. When it finishes, it should clean up after itself and delete all the temporary files. If anything fails here, there may be temporary files and directories that need to be cleaned up manually. That's not a particularly major problem, but rather just a potential event that you should be aware of.

More complex setups

If you have everything contained within your Python code, you're done at this point. But for many programs, you will have some extra data or files that need to be part of the package which is sent out. For example, say you had a readme file that contained important information for the user. In these cases, you can include them with a command like the one below.

```
pyinstaller --add-data "src/README.txt:."
myscript.py
```

The format is `SOURCE:DESTINATION` for the `--add-data` option. In the above example, we are taking the readme file from the `src` subdirectory and putting it in the root distribution directory. If you have multiple files, you can include as many `--add-data` options as needed.



Above By default, PyInstaller builds a dependency graph for all the imported modules in your program

This is fine for data files and documentation, but what if you have some piece of optimised code in a shared library? In this case, you use the `--add-binary` option to add a list of files to be included in the final executable. It has the same format as the `--add-data` option, and you can also have multiple `--add-binary` options. With this option, you are telling PyInstaller to include those binary files in the list of files to package, but also to check these binary files to see what dependencies they have. This is where PyInstaller may have the most difficulty, so you will definitely want to test the executable being created before sending it out to the wider world.

> "If you have everything contained within your Python code, you're done at this point. But for many programs, you'll have some extra data or files that need to be part of the package"

Create spec files

For smaller projects, you can probably manage the executable creation with the options `--add-data` and `--add-binary` directly on the command line. This way of doing things will very quickly become untenable, however. Luckily, there is another way of organising everything that needs to be bundled together into the executable version of your program: the spec file. This file contains everything that you might want to configure about how your executable operates.

The first step is to create a spec file that you can edit for your project. The following command does just this.

```
pyi-makespec myscript.py
```

Rogue modules

Finding the modules being imported is one of the issues you may run into. You can add extra paths to the list being searched with the following option:

```
pyi-makespec
--paths=/
path/to/thisdir
myscript.py
```

To make this starting spec file as specific as possible to your project, you can include any command line options that you would have included when calling PyInstaller directly. See the following for a very simple, one-folder Python script:

```
block_cipher = None
a = Analysis(['myscript.py'],
                 pathex=['/home/jbernard/work/
myscript'],
                 binaries=None,
                 datas=None,
                 hiddenimports=[],
                 hookspath=None,
                 runtime_hooks=None,
                 excludes=None,
                 cipher=block_cipher)
pyz = PYZ(a.pure, a.zipped_data,
             cipher=block_cipher)
exe = EXE(pyz,... )
coll = COLLECT(...)
```

This script is run as code by PyInstaller, and ends up creating objects of the four classes Analysis, PYZ, EXE and COLLECT. The Analysis object checks for all the files that are meant to be included and generates the list. The COLLECT object actually puts copies of all the required files into the distribution directory. The EXE object generates the executable file that will be used to run the program. If you are sending out a ZIP file containing the distribution folder, this is where PyInstaller stops. If, instead, you generate a single executable file, PyInstaller uses the PYZ object to handle this step.

> “There is a lot more you can do during the generation of your executable. For example, you can add command line options to Python to be used when it runs your program”

Setting environments

If you need to do some environment setup for your project, you can add the following option to pyinstaller.

```
--runtime-
hook=file1.
py --runtime-
hook=file2.py
```

These Python scripts are run, in the order given, before your program is run.

As you can see in the above example, there are several entries to manage the creation of your stand-alone application. The entries **datas** and **binaries** are equivalent to the two command line options we looked at earlier, `--add-data` and `--add-binary`. So for example you could use the code below to add a readme file to your packaged application.

```
datas=[('src/README.txt', '.')]
```

As you can see, the list of files is simply a Python list. This means you can easily add a large number of files without too much trouble.

There is a lot more you can do during the generation of your executable. For example, you can add command line options to Python to be used when it runs your program. The following code adds the Python command line options `v` (to write a message to stdout each time a module is initialised) and `W ignore` (an option to change warning behaviour).

```
options = [ ('v', None, 'OPTION'), ('W ignore',
None, 'OPTION') ]
a = Analysis( ...
                )

In order to use these options, you need to
add the following to the creation of the EXE
object:

exe = EXE(pyz,
            options, ...
            )
```

For when your project actually involves several different Python programs that it makes sense to package together, PyInstaller includes a function named `MERGE`. You can create a spec file where you define a series of Analysis objects, one for each program. You then define a `MERGE` entry, listing each of the Analysis objects that should be processed.

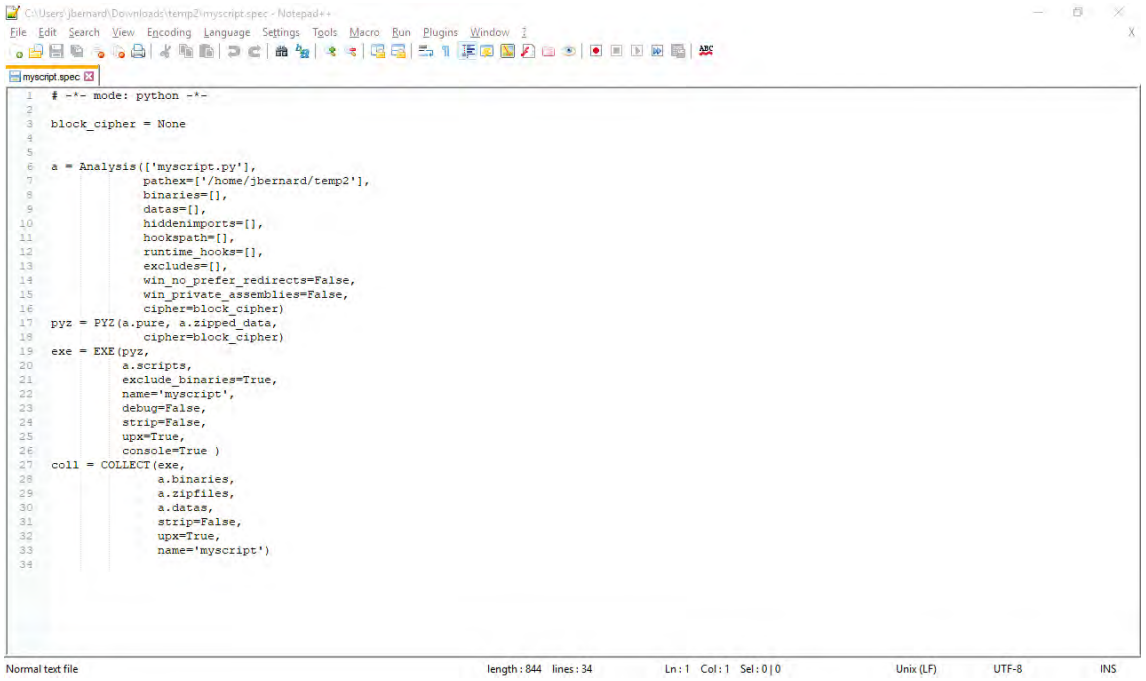
Check for problems

Because the Python environment is large, complex and ever-changing, you will inevitably run into issues where PyInstaller fails to generate a working executable version of your program. Luckily, it provides some tools that you can use to debug these problems.

The first option is to look at the output file where all warning messages are written. PyInstaller creates a subdirectory called **build** where the program is collected and built. Within this subdirectory is another subdirectory named after your script. So in our example, the warning file would be named `./build/myscript/warnmyscript.txt`. All the warnings from PyInstaller are written here, such as messages like “module not found”. If you want to see these messages on the console, you can use the command line option `--log-level`.

The dependency graph

A second tool available to you is the dependency graph. An HTML version of this is built automatically by PyInstaller. You can find it in the file `./build/myscript/xref-myscript.html`. This file lists the contents of the import graph; you can navigate around it by clicking the ‘imported by’ labels. If you use the `--log-level` command line option and set it to `DEBUG`, PyInstaller also generates a GraphViz file called `./build/myscript/graph-myscript.dot`. This file can then be run through GraphViz to generate a graphical representation of the dependency graph. This file can get rather large, since there are so many imported Python modules, even in the simplest of programs.



Left PyInstaller starts by creating a spec file. You can edit it to add everything you need

These two tools will help you to identify what PyInstaller is looking for, and what it is having difficulty finding. There are also a number of special cases that have already been identified, such as building django files or packaging multiprocessing programs. They are available under the recipes section of the PyInstaller documentation site.

Use freeze

PyInstaller is not the only option available to package Python programs. Two other popular options are freeze and py2exe. Freeze actually comes in a couple of implementations, cx_Freeze and bbFreeze, although the latter hasn't been updated in quite a while, so it might be more trouble than it's worth.

cx_Freeze produces results similar to the distribution folder output from PyInstaller. Running the command:

```
cxfreeze myscript.py --target-dir dist
```

produces an executable file, plus all required libraries, within the dist subdirectory. If you want to distribute this to end users, you need to zip up the target directory and have the user unzip it locally. One part that is different from PyInstaller is that cx_Freeze doesn't handle external data files as cleanly. You would need to add some extra code to manage locating them, as discussed in the cx_Freeze documentation.

Use py2exe

py2exe is a very different option to cx_Freeze and PyInstaller. py2exe is actually an extension to distutil. This means that you need to create a **setup.py** file in order to generate an executable version of your program. Using our example above, the following would be a basic setup file:

```
from distutils.core import setup
import py2exe

setup(console=['myscript.py'])
```

As you can see, the **setup.py** file imports both distutils and py2exe. You can then run this file with the following:

```
python setup.py py2exe
```

This process generates a huge amount of output, and also creates two subdirectories called **dist** and **build**. **build** is the working location where all the work of creating the executable happens; **dist** contains the final form of the executable. In order to test it, enter the **dist** subdirectory. You can then test it with the command **./myscript**. To distribute this executable, zip up the dist subdirectory, as with cx_Freeze.

With both of these programs, dealing with special cases is more difficult than with PyInstaller. If your project is basic, any of the executable creation tools should be sufficient. If your project is more complex, however, or uses a fair amount of external data and/or code, PyInstaller might be your better option.

Sharing Python code has always been an issue. The combination of having an interpreted language, along with a huge collection of third-party modules, is usually a powerful environment for program development. However, this very same strength becomes a weakness when you want to share your work with others. Hopefully, this article has presented some ideas that you can use in order to share your Raspberry Pi projects easily.

PART THREE

Go: Add a database to your video-sharing server

This issue, we'll integrate an SQL database with our video sharing server to enable keyword searching

John Gowers

John is a university tutor in Programming and Computer Science. He likes to install Linux on every device he can get his hands on and has extensive programming experience.

Resources

Go
Download through your package manager or from **https://golang.org/dl**

A web browser
Ideally Chrome 4.0+, Firefox 3.5+ or Opera 25+

MP4 video files
One or more MP4 for testing

Sample code
from **LU&D195** on the coverdisc or FileSilo

```
100    sqlQuery := `SELECT *,
101     (SELECT COUNT(ID) FROM Keywords
102      WHERE VideoID = Videos.ID
103      AND Word IN (?` + strings.Repeat(",?", len(keywords) - 1) + `))
104     AS Matches
105    FROM Videos
106     HAVING Matches > 0
107     ORDER BY MATCHES DESC`
108
109    interfaceKeywords := make([]interface{}, len(keywords))
110    for i := range keywords {
111      interfaceKeywords[i] = interface{}(keywords[i])
112    }
113
```

Above Go includes a comprehensive library for communicating with databases using the SQL language. Here, we have mixed in some Go code with the SQL in order to programmatically build up the query

Over the last two issues, we've been building up a web server that enables people to upload videos with titles. The only slight limitation is the ability to browse videos. The current implementation puts a long list of all the uploaded videos on to the home page, but this is clearly going to be impractical if the site becomes popular enough. A better approach is to implement some kind of search, so that people can more easily find the videos they're interested in.

It's not hard to think of ways we could do this using our existing infrastructure: for example, by saving a list of keywords into our `videodata.txt` files, and perhaps maintaining some form of index in order to help speed up search using the file system.

A better approach, however, is to realise that our current setup is inadequate for what we want it to do, and that we should use a tool designed for storing and retrieving data in this way: an SQL database.

If you're joining the series this issue, you can find a complete set of code on the coverdisc and the FileSilo. We will use this code as the starting point for this article.

Set up the database

The first step is to set up the database. If you have a MySQL implementation installed already, you can use that. Otherwise, we will be using MariaDB, a free software

An introduction to SQL

SQL, the Structured Query Language, is an almost universally used language for speaking to relational databases and gathering information from them. A database is made up of tables – grids, like the one shown in **Figure 3** – and SQL is a very expressive language for gathering information from these tables. Besides the bare data tables, we can form other tables, called queries, whose values are computed from the existing tables, by writing appropriate SQL statements.

The most basic SQL keyword is `SELECT`, which gets information from a number of columns. So `SELECT * FROM Videos;` gets us the entire Videos table, while `SELECT Title FROM Videos;` gets us just the `Titles` column. We can place constraints on these queries using the `WHERE` keyword: for example, `SELECT ID FROM Videos WHERE Title = 'Trip to the zoo';` gets us a list of IDs of videos with the given title.

Another command we use is `COUNT`: enclosing a query inside `COUNT` collapses the resulting table to its number of columns. For example, `SELECT COUNT(ID) FROM Videos;` will get us the number of videos in our collection.

fork of MySQL. Once it's installed, you should run the following command to perform some intial setup:

```
# mysql_install_db --user=mysql --basedir=usr
--datadir=/var/lib/mysql
```

After this, start the MariaDB service with `service mysql start` or `systemctl start mariadb` depending on your distribution. You can now bring up the MariaDB prompt by running the following command in a command window:

```
$ mysql -u root -p
```

You should be prompted to enter the password for the root SQL user. If you've just installed MariaDB, a blank password should work; you can change this later. To start with, we'll create a database called `lutube` and a user, also called `lutube`, who will be responsible for administering it:

```
> CREATE DATABASE 'lutube';
> CREATE USER 'lutube'@'localhost';
> GRANT ALL PRIVILEGES ON 'lutube'.* TO
'lutube'@'localhost';
```

Then run `\q` to close this instance. Re-run MariaDB using the `lutube` user by running the command `mysql -u lutube`. Run the command `USE lutube;` to switch to the database we have created. First, we will create a table to store information about all the videos on the site:

```
> CREATE TABLE Videos (
>    ID varchar(100) NOT NULL PRIMARY KEY,
>    Title varchar(200)
> );
```

This creates a table with two columns, ID and Title, which we should be familiar with already. Keep this window open so we can see how the database changes as our program operates on it.

Integrate the database with Go

Go programmers can interact with SQL databases using the `database/sql` package. However, since this package is intended to work as generally as possible, it does not provide any code for dealing with individual database implementations. We'll download the Go MySQL driver, since MariaDB is a MySQL implementation. We can do this by running the following command:

```
$ go get github.com/go-sql-driver/mysql
```

In order to use this package, we need to add the following two imports into the `import` statement at the top of `lutube.go`:

```
"database/sql"
_ "github.com/go-sql-driver/mysql"
```

Do not forget the underscore, which means that we are importing the package for its side effects only. Go interacts with the database through an object of type `sql.DB`. To avoid having to pass this around a lot, let's make it a global variable by putting the following declaration outside the scope of any function.

Figure 1

```
func openVideoDatabase(databaseName string) error {
   var err error
   videoDatabase, err = sql.Open("mysql", databaseName)
   return err
}
func closeVideoDatabase() {
   videoDatabase.Close()
}
```

Above The global variable `videoDatabase` will be the object through which we perform all communication with the database

```
var videoDatabase *sql.DB
```

Now let's write some functions for setting up this database, and for closing it again. Sample functions are shown in **Figure 1**. The string `"mysql"` that we pass into `sql.Open` refers to the particular SQL driver package that we have chosen.

We now call these functions at the very start of our `main` function:

```
openVideoDatabase("lutube@/lutube")
defer closeVideoDatabase()
```

The string `lutube@/lutube` refers to the database we created earlier. You should deal with the error value returned from `openVideoDatabase`, perhaps by passing it to `panic` if it is not `nil` and immediately returning.

> "To get to grips with Go-SQL integration, we'll write a short tool to migrate our existing videos"

Work with SQL statements

In order to get to grips with the Go-SQL integration, we'll write a short tool to migrate our existing videos to the new SQL system. Since this will be a separate program, we'll need to put the common functions into a separate file. It makes sense to have a new file to hold all the functions used for storing and reading videos, so create a new file called `videos.go`, and move into it the two new functions `openVideoDatabase` that we have just written, plus the functions `loadVideo`, `saveVideo` and `getAvailableVideos` that were already there. Also move the `videoDatabase` global variable and the definition of the `Video` struct into `videos.go`. Put `package main` at the top of the file, as in `lutube.go`.

This will mess up the package imports a bit: we still need to write an `import` statement for `videos.go`, and

there are some imports in `lutube.go` that are only used in the functions we have moved across, so these need to be removed. The easiest way to see what needs to go and what can stay is to try to compile the original program using the command:

```
$ go build lutube.go videos.go
```

and to look at the error messages. You should also move the import of the SQL driver `github.com/go-sql-driver/mysql` from `lutube.go` into `videos.go`.

The first step for our migration tool is to add another function in `videos.go` called `saveVideoToDatabase`, which will take a video and store its ID and title in the `Videos` table that we created earlier. First write the function header:

```
func saveVideoToDatabase(video *Video) error
{
    // body goes here
}
```

In order to communicate with the database, we use the `Exec`, `Query` and `QueryRow` functions from the `sql` package. We use `Exec` when we want to execute an SQL statement and don't expect to get any information back, `QueryRow` when we expect to get back a row of data and `Query` when we expect to get multiple rows.

Since the `saveVideoToDatabase` function does not need to get any information out of the database, we use `Exec`. We write the body of the function as follows.

```
_, err := videoDatabase.Exec("INSERT INTO
Videos VALUES (?, ?)", video.Id, video.Title)
return err
```

The string that we pass into `Exec` is an SQL statement for inserting values into a table, and the two question marks are placeholders that will be filled with the values `video.Id` and `video.Title` that we have passed into the function. If you are unfamiliar with SQL, don't worry – we'll be providing all the commands that you need and explaining them as we go.

Use the database

All we need to do now is to write the migration tool. For this, we create a new file called `migrate.go`, and copy in the `main` function from **Figure 2**, as well as the `package main` declaration at the top. The `main` function uses the functions we've written to iterate over the videos on the file system, saving them all to the new database. Compile the program with `go build migrate.go videos.go` and run it with `./migrate`.

You should be able to check that this has worked by going back to your MariaDB window and typing the command `SELECT * FROM Videos;`. If everything has worked, you should see a table as in **Figure 3**. This means the videos have migrated successfully to the database.

We need to modify the behaviour of the server so that it uses the new database, rather than the file system-based setup we had before. To do this, we will need to rewrite a number of the functions in `videos.go` (a side effect of this is that our migrate tool will no longer work, but that's fine).

Let's start with the `loadVideo` function. First, delete the existing contents of the function: we're going to rewrite it completely. `loadVideo` takes in a parameter, `id`, for the ID of the video to load. The first step is to run an SQL statement that will return us the corresponding row of the table:

```
row := videoDatabase.QueryRow("SELECT * FROM
Videos WHERE ID = ?", id)
```

Next, we create a new `Video` object and use this row to populate it:

```
var video Video
err := row.Scan(&video.Id, &video.Title)
```

Deal with the error in the usual way. Lastly, we can return (a pointer to) the video, completing the function:

```
return &video, nil
```

When rewriting the `saveVideo` function, we want to keep

Figure 2

```
func main() {
    openVideoDatabase("lutube@/lutube")
    defer closeVideoDatabase()
    availableVideos, _ :=
    getAvailableVideos()
    for _, video := range availableVideos
    {
        saveVideoToDatabase(video)
    }
}
```

Above With the functions we have written, it's possible to write a tool to migrate our existing video library with only a few lines of code

Figure 3

```
MariaDB [(none)]> USE lutube;
Database changed
MariaDB [lutube]> SELECT * FROM Videos;
+-----------+----------------------+
| ID        | Title                |
+-----------+----------------------+
| 383618752 | Carousel             |
| 389996965 | Richard at the beach |
| zoo       | Trip to the zoo      |
+-----------+----------------------+
3 rows in set (0.00 sec)

MariaDB [lutube]>
```

Above MariaDB's command line interface makes it easy to view the contents of database tables or queries

SQL injection attacks

You might wonder why we bother with the ? placeholders, particularly since we are able to programmatically construct the SQL statement inside Go. The reason is to protect our code from what is called an SQL injection attack. Inserting user-provided text directly into our SQL queries could allow our users to execute their own SQL code on our database.

the first few lines, which deal with saving the video file to the file system. But we should delete the subsequent lines, which save the video's title into the `videodata.txt` file. Instead, we want to save the video to the database. Luckily, we've already written a function to do this, so we can just write:

```
err = saveVideoToDatabase(video)
return err
```

The last function that you need to change should be `getAvailableVideos`. Previously, we implemented this function by searching for directories in the `videos` folder, but it is more robust to get them from the database. Once again, clear out the existing contents of the function – we're going to rewrite it completely.

First, we use the `Query` function to send an SQL statement that will return the list of all the rows in the `Videos` table:

```
rows, err := videoDatabase.Query("SELECT * FROM
Videos")
```

Deal with the error as you see fit. This query will return the list of all the IDs of videos in the table.

`getAvailableVideos` should return a slice of pointers to `Videos`. First, we create an empty slice (as in the original implementation of this function):

```
availableVideos := make([]*Video, 0)
```

We are now going to iterate over the object `rows`, using a `for` loop. For each iteration of this loop – that is, for each row of the table – we create a new `Video` object and use the row of the table to fill in its fields, before adding the video to the list:

```
for rows.Next() {
   var video Video
   err := rows.Scan(&video.Id, &video.Title)
   availableVideos = append(availableVideos,
&video)
}
```

Last, we return the new list of available videos:

```
return availableVideos, nil
```

Recompile the code with `go build lutube.go videos.go`. If you have done everything right, you should get no errors, apart from a message telling you you need to remove the import `io/ioutil` from `videos.go`. Remove this import and then recompile the program.

Run `./lutube`, then fire up `http://localhost:8080` in a browser and check that you are still able to play and upload videos, and that uploaded videos appear on the home page when you reload it. If everything is working as before, then we can move on to the next step. Otherwise, check your working and any error-handling code you may have introduced.

> “We need to modify the behaviour of the server so that it uses the new database, rather than a file system-based setup”

We want to be able to search for videos by keywords: each video will have a number of keywords associated with it, and we should be able to search for them and be presented with a list of all the matching videos.

Keyword search

The first step is to create a new table in our `lutube` database. Inside your MariaDB command window, run the command shown in **Figure 4** to create the table.

This table will have one row for every keyword from each video on the server. In particular, each video will have multiple rows in the table corresponding to it, one for each video keyword, and each keyword will occur in multiple rows of the table, once for each video where it appears. The `ID` field is a convenient counter used to uniquely identify rows of the table, and the `FOREIGN KEY` specification ensures that the `VideoID` field is indeed the ID of a video living in the `Videos` table.

We haven't written any functionality to add keywords to videos through the web interface, so spend some time adding them through the command-line MySQL interface instead. For example:

```
> INSERT INTO Keywords (Word, VideoID)
>     -> VALUES
>     -> ('elephant', 'zoo'),
>     -> ('flamingos', 'zoo'),
>     -> ('carousel', '383618752');
```

Use `SELECT * FROM Keywords` to look at your handiwork when you have finished.

Now we can start implementing the search. First, we write the function (in `videos.go`) that's going to take the

Figure 4

```
CREATE TABLE Keywords (
   ID int PRIMARY KEY NOT NULL AUTO_
   INCREMENT,
   Word varchar(200),
   VideoID varchar(100),
   FOREIGN KEY (VideoID) REFERENCES
   Videos.ID
);
```

Above The `FOREIGN KEY` constraint is a useful sanity check to make sure that the `VideoID` column only contains valid video IDs, so we don't get tripped up

Figure 5

```
SELECT *,
 (SELECT COUNT(ID) FROM Keywords
  WHERE VideoID = Videos.ID
  AND Word IN ('elephant', 'flamingos',
  'carousel'))
 AS Matches
FROM Videos;
```

Above We can add temporary extra columns to a table that are computed from existing columns, such as the Matches column shown here

Figure 6

```
+-----------+---------------------+---------+
| ID        | Title               | Matches |
+-----------+---------------------+---------+
| 383618752 | Carousel            | 1       |
| 389996965 | Richard at the beach | 0       |
| zoo       | Trip to the zoo     | 2       |
+-----------+---------------------+---------+
```

Above Output from the statement in **Figure 5**. We can use the ORDER BY keyword to rank rows by the number of matches if we want to

list of keywords and convert it into search results. The function will take in a list of keywords and return a slice of pointers to appropriate **Video** objects:

```
func getMatchingVideos(keywords []string)
([]*Video, error) {
    //...
}
```

The first thing this function needs to do is to create a new slice to hold the videos:

```
videos := make([]*Video, 0)
```

Next, it should build up the SQL query that will get us the search results. One slightly complicated bit is that the number of keywords is variable, so we need to add some code to add a variable number of question marks into the query! See the main image on the first page for an example (lines 100 to 107). In this bit of code, we've used backticks `...` rather than inverted commas "..." for string delimiters, which allows us to split a string across multiple lines. We need to import the "strings" library in order to use strings.Repeat, which we have used to repeat the string ,? for every keyword from the second to the last, giving us a string like ?,?,?,?.

One other thing we need to do is to use casting to convert our list of strings into a list of interface{} objects, since the Query function expects arguments of this type:

```
interfaceKeywords := make([]interface{},
len(keywords))
for i := range keywords {
   interfaceKeywords[i] = interface{}
(keywords[i])
}
```

We can now call the Query function to get the rows of the table corresponding to videos that match the keywords:

```
rows, err := videoDatabase.Query(sqlQuery,
interfaceKeywords...)
```

The ellipsis ... after keywords is because Query expects a variable number of arguments, whereas we have our keywords stored in an array or slice.

After dealing with the error, we use a for loop to populate the videos slice. See **Figure 7** for an example, with error-handling stripped out. We can return it with `return videos, nil`.

Upgrade the web interface

The next step is to integrate all this with the web interface. Open the HTML template home.html and add a short form on it for carrying out searches:

```
<form id="search-form" action="/search/"
method="post">
  <input type="text" name="keywords" />
  <input type="submit" value="Search LuTube"/>
</form>
```

We will need to create a new HTML template called search_results.html, which will print out a list of

How our SQL statement works

The SQL statement shown in the main image might look a bit daunting, but it becomes easier to understand once we break it up into its constituent parts. First, consider the simpler statement shown in **Figure 5**, which produces output like that shown in **Figure 6**. Everything between the brackets (between the second and fourth lines) could be a standalone SQL statement, as long as we replaced Video.ID with an actual Video ID, such as 'zoo'. This statement would then return the number of keywords from the set 'elephant','flamingos','carousel' that match the 'zoo' video.

Now, if we put this whole expression in brackets and treat it as though it were a column in the Videos table, we can get the number of matches for *every* video in the table by inserting Videos.ID in place of 'zoo'. Adding AS Matches at the end gives a name to this temporary computed column – Matches, completing **Figure 5**.

It's not too hard to get from here to the code in the main image. The ORDER BY constraint tells the database to order the results by the number of matches, while the HAVING clause removes all rows where the number of matches is 0. Last, we replace the actual keywords with a list of placeholders (for example, ?,?,?).

Since the number of keywords might change, we need to write some Go code that will generate the right number of question marks for these placeholders.

Figure 7

```
for rows.Next() {
   var video Video
   var matches int
   rows.Scan(&video.Id, &video.Title,
   &matches)
   videos = append(videos, &video)
}
```

Above We have to create a dummy variable, matches, to store the number of matching keywords. You might like to save this variable and display it on the results screen

> An easy way to add keywords for a video automatically is to extract them from the video's title at upload time

matching videos. The code on this template is very similar to the code that prints out the list of all the videos on the home page; take a look at **Figure 8** for an example implementation.

Last, we need to handle the search itself. The form we just added sends its request to the `/search/` URL, so we should write a new handler function called `searchHandler` inside `lutube.go` to deal with that:

```
func searchHandler(writer http.ResponseWriter,
request *http.Request) {
   // ...
}
```

First, this function should parse the form in order to be able to read the data:

```
err := request.ParseForm()
```

After dealing with the error value, we can then get the keywords that were typed on to the form:

```
keywordList := request.FormValue("keywords")
keywords := strings.Split(keywordList, " ")
```

We have used `strings.Split` to split up the string that the user typed into individual words. This means that we have to import the `"strings"` library into `lutube.go`.

Next, we pass these keywords into our `getMatchingVideos` function, using the returned list of `Videos` to render the `search_results` template. Stripping out error handling, we end up with code that looks something like this:

Figure 8

```
<h1>Search results</h1>
<ul>
   {{range $a = .}}
   <li><a href="/watch/{{a.Id}}">{{a.
   Title}}</a></li>
   {{end}}
</ul>
```

Above Go's useful HTML template library makes it easy to display a list of search results that link to the appropriate videos, like this

```
matchingVideos, _ :=
getMatchingVideos(keywords)
renderTemplate(writer, "search_results",
matchingVideos)
```

Lastly, we need to register this handler to the `search` action by adding a new line of code to the `main` function:

```
http.HandleFunc("/search/", searchHandler)
```

You can add proper error handling through one of our special 'handler' functions from the last issue if you feel comfortable with that.

Finish off

There's just one last thing to do: currently, we are only able to add keywords to the video library by interacting directly with the MariaDB console. It would be nice if we could insert keywords automatically when uploading videos. The easiest way to do this is to split up the video title into words and add them to the database, which we can do by adding the code in **Figure 9** into the `uploadHandler` function, just before the call to `http.Redirect` right at the end.

Compile with `go build lutube.go videos.go` and fire up `http://localhost:8080` in a web browser. If your code is all working, you should now be able to search for videos using the keywords stored in our SQL database.

Hopefully this series has given you an insight into how Go works as a language, as well as providing one solution (out of many, many approaches) to building a working video server. If you go even further and add extras, let us know – we'd love to see the results!

Figure 9

```
keywords := strings.Split(title, " ")
for _, keyword := range keywords {
   videoDatabase.Exec("INSERT INTO
Keywords (Word, VideoID) VALUES (?, ?)",
                        keyword, id)
}
```

Above The easiest way to populate the Keywords table is to insert all the words from the title when the video is uploaded

Further directions

We've only had room to describe how you might implement the most basic functionality in the server, and we'll leave it up to you to improve it. You might want to enable people to include a description alongside their video, or you might want to let them write their own list of keywords.

PICK THE PERFECT IDE

Linux provides a powerful platform for software developers and offers a wide choice of Integrated Development Environments, as **Paul O'Brien** explains...

AT A GLANCE

- Traditional full IDEs p58
  Full-fat IDEs remain popular, particularly in the Java space, courtesy of the Apache and Eclipse foundations
- Editor-based IDEs p60
  Many text editors, both full apps and command line, have gradually grown into full-blown IDEs
- The JetBrains behind the operation p61
  If you're willing to pay for your IDE, you can have access to a family of hugely impressive software development products
- Source-control tools p62
  The latest source-control tools bring visualisation to the process, including some additional superpowers

While adoption of Linux is gradually growing among general consumers, it remains a popular choice for one particularly tech-savvy group of users: software developers. The technical nature of the platform, which can scare away the average user, is something that's relished by developers as it provides ultimate customisability and an excellent level of performance for coders.

Software development on any operating system typically takes place within an Integrated Development Environment (IDE). An IDE is so named because it typically brings together the core functions needed for software development – such as a source code editor (always!), build tools and a debugger. Other elements can include built-in source control, GUI design, object-orientated development tools and intelligent code completion to make the development process easier.

When choosing an IDE, your selection is usually driven by the programming languages that are supported. Many IDEs choose to focus on a specific language and this has been the traditional approach in the past, but more recently IDEs have moved towards multi-language support, often provided by plug-ins created by the community. This trend has been driven by the growth in popularity of text editors that have grown into more fully featured IDEs; some of the most frequently used IDEs including Visual Studio Code, Atom and Brackets have humble origins as relatively basic editors.

Some of the most widely used IDEs are paid-for products, including the hugely popular Jetbrains suite, but many, particularly those that thrive on their open source extensions, are available for free. Many IDEs are also cross-platform, providing support for Linux, macOS and Windows. Even Microsoft's full-fat version of Visual Studio, typically a Windows-only product, now has a Mac version and a Linux version seems likely soon, thanks to the newly open Microsoft and its desire to become more platform-agnostic.

> "IDEs have moved towards multi-language support, often provided by plug-ins created by the community"

When developing for Linux itself, some IDEs will be environment-specific. GNOME Builder is a general-purpose IDE specifically tailored for writing software for the GNOME desktop environment. It has integrated support for DevHelp, Git and Flatpak, with code completion for the C languages and a plug-in-based architecture. On the KDE side of the fence, Qt Creator provides a tailored IDE for those developing using KDE libraries, as does KDevelop, another development environment that forms part of the KDE project.

As one might expect, there are frequent debates around what is the best development environment to use for Linux kernel development. Vim is typically the preferred choice, reflecting the more technical nature of kernel development. Let's look at some of the options.

Pick the Perfect IDE

Do not adjust your sets – we really are starting a feature about Linux IDEs with a Microsoft product! Visual Studio Code (**https://code.visualstudio.com**) is an open source source code editor from Microsoft that is available for Linux, Windows and macOS. Code was announced in April 2015 by Microsoft at its Build developer conference and released as a preview shortly afterwards, before being fully released in April 2016. As with many cross-platform projects, Code is based on the node.js-based Electron framework, which is used together with the Chromium Blink layout engine to deploy desktop Node applications.

Code supports a huge number of programming languages with varying levels of support for the four core features of the IDE: snippets, advanced syntax highlighting (aka IntelliSense), brace matching and code folding. The feature set of Code is growing all the time courtesy of open source extensions, which include code analysis/linting tools courtesy of support for the Language Server Protocol (LSP), which was originally developed for Code but is now used by several other IDEs.

IntelliSense is the real jewel in the crown of Visual Studio Code. As well as the all-important basic syntax highlighting and auto-complete, Code has additional intelligence which provides contextual 'smart' completions based on elements of your code, including variable types, included modules and other functions within your code. Code integrates with Git and numerous other SCM (Source Code Management) providers to offer the full range of available functionality in the editor, including diff review, staging, commit, push and pull.

> "Code supports a huge number of languages with varying levels of support for the four core features of the IDE"

There are a truly vast number of extensions for Code available in the marketplace. The most popular extensions provide full Python and C/C++ support, closely followed by JavaScript debugger integration with Chrome and support for ESLint, an enhanced JavaScript linting tool. As an indication of the popularity of Code, all the top 10 extensions have had over 5 million downloads, and well over 50 have had over 1 million downloads.

QUICK FACT

Preview Code versions

As well as the stable release, Microsoft offer an Insider's Edition of Visual Studio Code. This enables you to check out the newest features at least every day, side by side with the stable version, albeit with the added risk of it being broken!

Atom

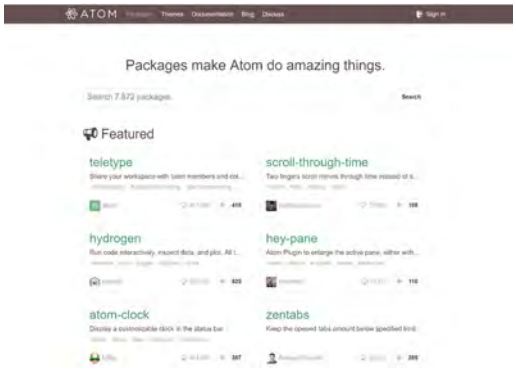
Often mentioned alongside Visual Studio Code is Atom (**https://ide.atom.io**). Like Code, Atom is another free and open source editor for Linux, Windows and macOS based on Electron. Atom preceded Microsoft's offering and is developed by Git heavyweight GitHub (ironically, itself now in the process of being purchased by Microsoft).

While originally pitched as an editor, Atom is now also bundled as 'Atom IDE', which includes a selection of add-on packages (the Atom name for plug-ins or extensions) to provide auto-completion, improved code navigation courtesy of an outline view, definition jump, hover preview and reference search features.

Atom's speciality is a feature that Code has also recently added: online collaboration. Branded as 'Teletype' in Atom, the feature allows developers to share

QUICK GUIDE

Visual Studio Code power tips



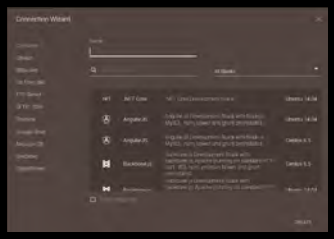
Above Use the Atom plug-ins site to add additional functionality for the Editor; themes can also be downloaded and installed

If you're a Visual Studio Code user, we have a few tips for you on how to make the most of the editor. Our first tip involves the command palette. Think of it as 'terminal just for Code' – it enables you to access various commands just by typing them out, which is, of course, far quicker than using the mouse. To open the palette, use **Ctrl**+**Shift**+**P**. Just start typing to filter a list of commands.

Another feature that will be familiar to users of full-fat IDEs is 'Go to definition'. This is supported in Code too. Simply highlight a variable or method and press **F12** to jump to the definition, or **Alt**+**F12** to peek rather than jump. One of our favourite Visual Studio Code features is the ability to edit multiple lines at once, enabled with the use of multiple cursors. Hold **Alt** and click on additional locations to create extra cursors and then go ahead and make your changes. The feature is incredibly powerful.

Web-based IDEs

Don't fancy installing yet another set of packages? Try these browser-based solutions instead



1 Coder

https://coder.com

Brand new and available as a free Public Alpha, the Coder web-based IDE enables you to develop alone or with others in 15 languages, with features found in full IDEs including autocomplete, syntax highlighting, linting and go-to-definition. Code is compiled, tested and run in your own scalable container.

2 AWS Cloud 9

https://aws.amazon.com/cloud9

Cloud 9, which is now part of Amazon Web Services, is a cloud-based IDE code editor, debugger and terminal that comes packaged with tools for several popular languages including Python and PHP. Cloud9 is tailored towards serverless development using the Amazon suite of products and enables pair programming with a shared development environment.

3 Codeanywhere

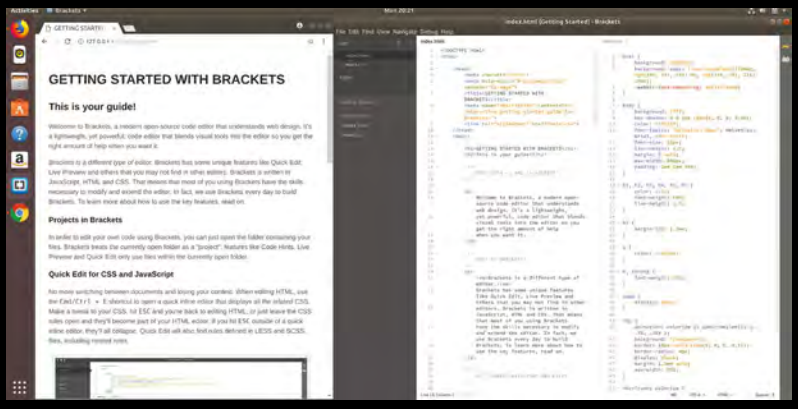
https://codeanywhere.com

Codeanywhere provides a customised development environment for an incredible 72 languages, with code completion for web languages, linting, multiple cursors, code beautifying and wide device and browser support. Code can be edited remotely whether it's hosted on FTP, SFTP or even Dropbox or Google Drive. A built-in terminal console is included on a provided container.

4 Codenvy

https://codenvy.com

Codenvy, now owned by Red Hat, is built on open source cloud IDE Eclipse Che, with added multi-tenancy, security controls and automation. Codenvy enables developers to develop code with portable Docker runtimes. A 'dev mode' production adds Intellisense, debuggers and desktop sync.



Above Brackets uses the Chrome remote debugging feature to power a 'Live Preview', with code changes reflected as you type

their workspace with team members and collaborate on code in real time. Teletype is based around the concept of 'portals' for sharing workspaces. When a developer is designated as the 'host' and opens a portal, their currently active tab then becomes a shared workspace. Invitees can join in and make edits in real time. As the host moves between files, collaborators also move to the active tab automatically. Teletype is based on WebRTC and therefore is fully peer-to-peer – there is no centralised server.

Just as Code has its marketplace, Atom too has an online package site. Nearly 8,000 different packages are available, installation of which is as simple as searching in the Settings UI itself and clicking Install, or by using the apm tool from a terminal.

As you'd expect given the ownership, Atom has first-class integration with GitHub. This isn't at the expense of other source control services however – the same package that enables GitHub also integrates with other standard Git providers. Packages also provide support for alternative SCMs including SVN.

Brackets

Last in our cross-platform, editor-based IDEs list is Brackets (**http://brackets.io**). An open source tool developed by Adobe, Brackets is less popular than Code and Atom, but still has a loyal following. While the former support a broad range of languages, Brackets is very much tailored towards web development, focusing on live HTML, CSS and JavaScript editing. As you'd expect from an Adobe product, Brackets blends visual tools into the development experience and its signature features make it feel like quite a different proposition to the other heavyweights in this area.

Brackets uses inline editors to simplify the development flow itself. While working on an element in HTML, a quick press of a key provides the ability to modify the CSS for that element without jumping into the main CSS file. Changes that are made can be viewed instantly courtesy of Live Preview, which maintains a real-time connection to the browser. Live Highlight lets you see where styling is being applied live in the browser. Brackets feels like a powered-up version of the in-browser developer tools.

Vim, aka 'Vi Improved', is a souped-up clone of the Vi text editor, the line editor's dream. Vim is designed for use both from a command-line interface and standalone in a GUI. It's bundled on many Linux distros as standard and becomes an IDE courtesy of its extensive plug-in system, support for hundreds of programming languages and file formats and numerous integrations.

Let's be honest though – Vim is not for the faint-hearted. It has such a vast range of options, shortcuts and commands that most users will never discover all of them. It's highly customisable via the .vimrc file, so once you're up to speed and have it configured just how you like it, it's very fast to use.

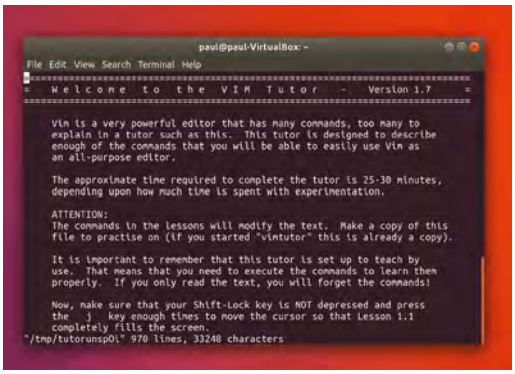
QUICK FACT

Coding .NET on Linux

.NET dev on Linux might not seem like a natural fit, but there are some great tools. JetBrains Rider is excellent, but Visual Studio Code and MonoDevelop are also worth a look. Omnisharp enables .NET development across a range of editors.

QUICK GUIDE

Getting started with Vim


paul@paul-VirtualBox: ~
File Edit View Search Terminal Help
Welcome to the VIM Tutor - Version 1.7
Vim is a very powerful editor that has many commands, too many to
explain in a tutor such as this. This tutor is designed to describe
enough of the commands that you will be able to easily use Vim as
an all-purpose editor.
The approximate time required to complete the tutor is 25-30 minutes,
depending upon how much time is spent with experimentation.
ATTENTION:
The commands in the lessons will modify the text. Make a copy of this
file to practise on (if you started "vimtutor" this is already a copy).
It is important to remember that this tutor is set up to teach by
use. That means that you need to execute the commands to learn them
properly. If you only read the text, you will forget the commands!
Now, make sure that your Shift-Lock key is NOT depressed and press
the j key enough times to move the cursor so that Lesson 1.1
completely fills the screen.
"/tmp/tutorunspOl" 970 lines, 33248 characters

Above When you master it, the speed of editing with Vim is unbeatable. Use the built-in tutorial to get you started

Using Vim can be daunting and even if you do decide to give it a try, it's easy to give up quickly because the learning curve is so steep. So, what is the best way to learn Vim and stick with it? A good place to start is Vim's own tutorial, launched with `vimtutor`. The tutorial will teach you everything you need to know to use Vim as an all-purpose editor, but of course can't cover everything! The tutorial is designed to take 25-30 minutes to complete. After you've mastered the tutorial, start using Vim in anger, without any plug-ins. You can use the .vimrc file to configure theming, syntax highlighting and other features to suit your preferences. After running like this for a period of time, you'll be ready to start dipping your toe in the water when it comes to plug-ins. Vim 8 has native plug-in package loading, but the process is still, of course, rather basic. Avoid getting too carried away initially, just install interesting looking packages individually. You'll have Vim tuned to your needs in no time!

Version 8 of Vim introduced a new package management feature which makes using multiple plug-ins a lot easier, and therefore running Vim as an IDE more feasible. Recommended plug-ins include 'project' for a project view of files, 'ide' for tracking file status, 'NERDtree' for a tree view, 'YouCompleteMe' as a must-have for code completion and 'gitv' for source control. Tracking down useful packages for Vim can be an effort in itself; however several third-party sites are available to help including **https://vimawesome.com**, which currently indexes an incredible 17,071 plug-ins.

If you're trying Vim for the first time, you might well be scared off at first glance, but give it a chance. It's incredibly useful to get to know, not least because it works over SSH connections, but also because once you have got the hang of it, the configuration is portable and there is a thriving community behind the app.

NetBeans

After souped-up text editors and the hardcore command-line tool, now we have what one might term a 'proper IDE'. NetBeans (**https://netbeans.org**) is primarily an IDE for Java that runs on Linux, Windows, macOS and Solaris, with modules included in the NetBeans IDE download for other languages including PHP, C, C++, HTML5 and JavaScript. NetBeans is free and open source; the application has had varied ownership through the years under Sun and Oracle, before being donated to the Apache Software Foundation in October 2016.

The NetBeans IDE is built using a modular system, with each module performing a specific function such as the built-in Java support, JavaScript editing functionality, profiling features, CVS version control and the GUI design tool formerly known as Project Matisse. As well as the additional modules included for language support, the NetBeans website includes a 'plug-in portal' from where additional functionality can be downloaded. The second most popular addition makes NetBeans look like Sublime Text. Ruby and Rails support and a mobile development toolset called Codename One also feature highly.

NetBeans 9, released late in July 2018, marks the first release of the IDE by the Apache Foundation. Key to the release is IP clearance of the Oracle code donation, together with support for Java versions 9 and 10. NetBeans at Apache is officially in an 'incubating' status.

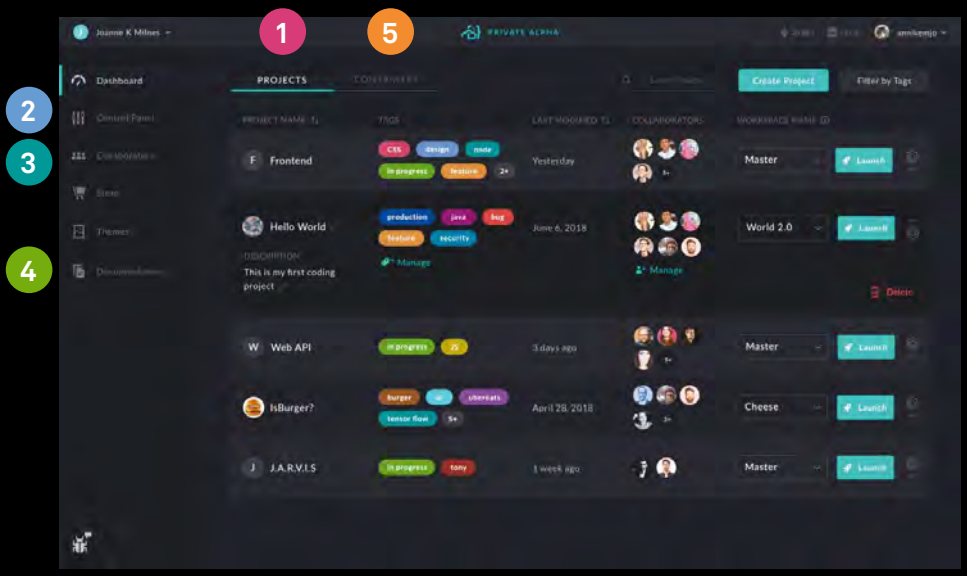
Eclipse

Like NetBeans, Eclipse (**www.eclipse.org**) is a free and open source IDE with its roots strongly in Java (it's the most widely used Java IDE), which now supports a wide range of development languages. Developed by the Eclipse Foundation on a codebase that originated from IBM in the form of VisualAge, the platform contains a base workspace and a plug-in system to enable

QUICK GUIDE

Using the coder.com web-based IDE

Web-based IDE users are still very much in the minority, but as the feature sets of the online products increase, they are becoming ever more viable for serious development. The Coder Public Alpha aims to be the best yet with an IDE that works on the most humble of hardware, including Chromebooks. Online IDEs take advantage of server processing power – Coder uses the concept of 'Fast Time', which dynamically scales your computational resources based on your needs with up to 96 CPU cores of power (at a price).



1 The dashboard contains a list of your projects, which can be assigned tags for filtering

2 The Control Panel enables you to configure your plan, purchase 'Fast Time' or storage, and view statistics

3 Web-based IDEs are all about collaboration. You can add and edit collaborators in the dedicated tab

4 Coder includes extensive documentation which includes a getting started guide and in-depth walkthrough

5 Containers are at the heart of the Coder platform. This tab lets you drop straight into your container shells

additional features. On the Eclipse site, several different downloads are provided to support different usage scenarios, so a developer doesn't have to download lots of plug-ins manually to get up and running with their required language.

As well as an Eclipse IDE for Eclipse committers (those developing the IDE itself), versions are offered for C/C++, Java, JavaScript and Web, Scientific Computing, PHP, Scout and a version specifically for testers. The latest 'Photon' release brings Java support to version 9 and 10, as with NetBeans, and includes PHP-specific improvements alongside a raft of underlying changes.

If you are choosing between NetBeans and Eclipse, opinion is very much divided over which is 'better'. NetBeans is generally considered to have a more accessible user interface and is often more quickly updated for new Java versions due to the close alignment to Oracle, while Eclipse has better third-party support and a broader plug-in ecosystem. More often than that the choice will be mandated by what's previously been used for the project you're working on and the third-party repositories you're going to connect to.

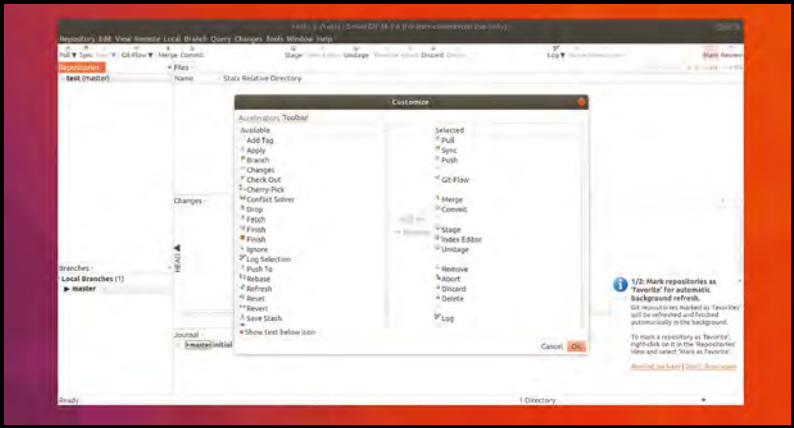
JetBrains

While all the options we have covered so far have been free, there is an elephant in the room when it comes to IDEs: JetBrains (**www.jetbrains.com**). Best known for its IntelliJ IDEA Java IDE, the company provides IDEs for a range of languages, typically via paid subscriptions but in some instances (such as IntelliJ) via free 'Community' editions too. Its products are cross-platform with Linux, Windows and macOS versions.

One of the key benefits of using a JetBrains IDE (apart from the fact they are just excellent products in their own right) is that after you've used one, you'll find that the user interface is consistent if you use another of their products. JetBrains also has a range of supporting products outside the IDEs themselves – it has created the Kotlin programming language, offers TeamCity for Continuous Integration and YouTrack for issue tracking. The technology behind IntelliJ IDEA also underpins Android Studio, provided for free by Google to enable Android development. Development for the mobile OS was previously enabled by an Eclipse plug-in; however the move to Android Studio has provided a far richer experience and provides a good way for JetBrains to expose developers to its products.

> "While all the options we have covered so far have been free, there is an elephant in the room when it comes to IDEs: JetBrains"

Pick the Perfect IDE



Above SmartGit is free for non-commercial use and is positively feature-packed, with extensive customisability

While most IDEs now provide the ability to integrate with source control (and of course, most source control systems work great from the command line), there are several source control tools which will make the development process easier.

GitKraken

GitKraken (**www.gitkraken.com**) from Axosoft is a Git GUI client. It provides a rich visual experience for viewing branching, merging and commit histories and optionally integrates with the leading online Git repositories. It's powerful and yet easy to use, with drag and drop support for complex tasks such as merging and rebasing. Version 4.0 of GitKraken includes its own code editor, ideal for quick tweaks and changes, with full support for syntax highlighting included. Advanced functionality includes git-flow support, submodules, Git hooks support and Git LFS compatibility. Free functionality is extensive, but a Pro licence adds improved merge-conflict resolution, multiple profiles and additional service integration. A fully installed Enterprise option is available with SSO support.

SmartGit

SmartGit (**www.syntevo.com/smartgit**) is a commercial graphical client for Git that also includes support for SVN. Like GitKraken, SmartGit includes drag and drop merging and rebasing, as well as commit re-ordering. The installation package includes everything needed to get up and running including Git itself, git-flow, an SSH client and compare and merge tools. While SmartGit's features largely match that of its competitors, it focuses on customisability, with highly configurable layouts, close integration with a range of external tools, extensive keyboard shortcuts, custom toolbars and (as with GitKraken) light and dark themes. The ability to create and resolve Pull Requests and Review Comments on GitHub and BitBucket is useful if that fits your workflow, as is the Distributed Reviews feature.

Git Extensions

Git Extensions (**https://github.com/gitextensions**) is a standalone UI tool for managing Git and SVN repositories that runs on multiple platforms courtesy of the Mono .net framework. It works well on Linux but is also particularly popular on Windows by virtue of its Windows Explorer integration and Visual Studio plug-in. The free tool is packaged in an installer that includes the app itself and the merge tool KDiff3. The user interface is simple but effective. The full range of Git functionality is supported, and Git Extensions is a good option for beginners, with an extensive manual and several video guides.

gmaster

gmaster (**https://gmaster.io**) is a brand new product which is currently free for non-commercial use. Pitched as the 'Git GUI with superpowers', gmaster aims to turn you into a diff and merge master by actually parsing your code. The app also helps you understand Git repos with unprecedent visualisation. It's developed by the same team behind 'Semantic Merge', which provides source code diff and merge based on language parsing, designed to deal with code that has been moved and modified. This deals with one of the biggest problems with working with large development efforts and large code merges: context. Semantic Merge and gmaster understand the code itself, which in turn helps you as the developer understand what's being added, removed or changed. Out of the box gmaster supports C#, C, C++, VB.net and Java. Additional user-contributed external parsers have already added support for Delphi, F#, Go and several others. The Linux release is 'coming soon' and it looks like it could be a game-changer.

QUICK FACT

Try multiple graphical UIs

Git doesn't limit what you can point at a repo. If you want to mostly use the command line but also have five different GUIs to the same location, you can!

Above The new code editor in GitKraken is perfect for making quick changes

QUICK GUIDE

Creating and using a new Git repo with GitKraken

After downloading and installing GitKraken, click the folder icon at the top left of the screen to create a new Git repo. The repo can be created locally, or online using the built-in integration with GitHub, GitLab, Bitbucket or Visual Studio Team Services. gitignore templates are included for a huge number of languages and an appropriate licence can be added automatically.

An initial commit will be created with the new files and the icons at the top of the screen can be used to push. At this point you don't even need to spark up your IDE – you can use GitKraken's built-in editor to start creating code! Select the tree view on the right-hand side of the screen, right-click and select 'Create file'. Double-click to open the editor.

UK subs turn to p30

NOT FROM THE UK?

SUBSCRIBE!

Don't wait for the latest issue to reach your local store – subscribe today and let *Linux Format* come straight to you!

3 GREAT WAYS TO SUBSCRIBE

Print, digital, and print + digital bundles!

» USA
From $15
every 3 months

» REST OF THE WORLD
From $15
every 3 months

» EUROPE
From €15
every 3 months

IT'S EASY TO SUBSCRIBE!

Visit: **www.myfavouritemagazines.co.uk/sublin**

Call: **+44 344 848 2852**

Lines open 8AM–7PM GMT weekdays, 10AM–2PM GMT Saturdays*

Savings compared to buying 13 full-priced issues. You'll receive 13 issues in a year. You can write to us or call us to cancel your subscription within 14 days of purchase. Your subscription is for the minimum term specified and will expire at the end of the current term. Payment is non-refundable after the 14-day cancellation period unless exceptional circumstances apply. Your statutory rights are not affected. Prices correct at time of print and subject to change. *UK calls will cost the same as other standard fixed line numbers (starting 01 or 02) and are included as part of any inclusive or free minutes allowances (if offered by your phone tariff) For full terms and conditions please visit http://bit.ly/magtandc. Offer ends 31 October 2018.

THE FUTURE OF MOORE'S LAW

Should we start getting used to a future of processor stagnation, with no expectation of speed improvements? **Mike Bedford** investigates

AT A GLANCE

- Explained: What is Moore's Law? p64
  We examine exactly what Moore's prediction states – and what it doesn't.
- Transistor scaling p65
  Why decreasing feature sizes is essential to increasing complexity.
- Shrinking issue p66
  We explain why the pace of shrinking transistor sizes has slowed.
- Light at the end of the tunnel p66
  Possible solutions to the end of Moore's Law.

Perhaps it seems like a very long time ago that we first started hearing about the impending end of Moore's Law; and in fact, a quick search reveals such predictions going back to 1997. We'll see later just what Moore's Law does and doesn't say, but the sentiment of these predictions was that a time will come when the relentless improvements in processor performance that we've grown to expect will come to an end. And when that time comes, we'll be faced with a stark realisation: this is as good as it gets.

Well, it seems you might have to start coming to terms with that outlook sooner rather than later because, according to some industry experts, Moore's Law has already come to an end. We did hope to get Intel's view on this question; after all, Gordon Moore, who gave his name to the law, was one of Intel's founding fathers. But while, historically, Intel has tended to wax lyrical about Moore's Law, on this occasion, the company declined to answer our questions on the subject. We'll leave you to come to your own conclusions about this. In the meantime, however, we'll present our own analysis of the future of Moore's Law and come to a view as to whether the long-standing merchants of doom are, for once, correct in predicting the end of an era.

What is Moore's Law?

Moore's Law has commonly been misquoted to say, for example, that it predicts a doubling in processor performance every couple of years. While this might not be too far from the mark, this prediction would be an implication of Moore's Law, as opposed to the substance of the law. In fact, when Gordon Moore made his famous statement, the first microprocessor was still six years away, the semiconductor industry was making discrete logic devices, and Intel was about to launch the world's first memory chips.

A 1965 article by Gordon Moore, published in the industry magazine *Electronics*, suggested that "the complexity for minimum component costs has increased at a rate of roughly a factor of two per year". He followed this by suggesting that "certainly over the short term this rate can be expected to continue, if not to increase. Over the longer term, the rate of increase is a bit more uncertain, although there is no reason to believe it will not remain nearly constant for at least ten years. That means by 1975, the number of components per integrated circuit for minimum cost will be 65,000."

If we remove the industry jargon, Moore was saying that the number of transistors in mainstream chips had doubled every year and he predicted that trend would hold true for at least ten years. He later amended this prediction to a doubling every two years, and that ten-year prediction has been extended over and over.

Although they didn't exist back in 1965, today it's reasonable to apply Moore's Law to microprocessors. The graph (starting p66) shows

the number of transistors per device from the first ever microprocessor, the Intel 4004, to the latest and greatest of a couple of years ago. We've concentrated on Intel chips, but that's not unreasonable since we're discussing an Intel prediction. Extending it to the current date requires a degree of speculation, because Intel no longer quotes transistor counts. The graph has a logarithmic vertical axis which means that any exponential growth – that is a doubling in a fixed period of time – appears as a straight line. The straight line shows the trend that would be expected if that doubling occurs every two years as predicted by Moore's Law.

You'll notice that in recent years most new products fell below that line, even though some of the more recent chips shown are high core-count, top-end Xeon chips. You might also notice that chips fell behind from around 1992 but, in time, the long-term trend was re-established. Naively we might assume that the current below-par performance might be another short-term glitch, but there's more to this than meets the eye.

Transistor scaling

While the graph of transistor counts doesn't seem to suggest that Moore's Law has hit the buffers quite yet, or at least not so badly that it can't recover, another trend does appear to be in trouble. That trend is the continual decrease in a chip's feature size.

> "Moore was saying that the number of transistors in mainstream chips had doubled every year"

If you take more than a passing interest in the chips that drive your PCs, you can't fail to have noticed that feature sizes decrease every few years, even if you're not entirely sure what that means. Indeed, if you do struggle to fully understand the term, you wouldn't be alone, because this has become a matter of fierce debate in recent years. This being the case, let's use a rather vague definition and say that the feature size is the size of individual features in a microprocessor – without getting embroiled in whether that's the size of a single transistor or a component part of a transistor.

When the 4004 processor first hit the streets in 1971, the feature size was 10 microns – which, if we express it in the unit more commonly used today – is 10,000nm. At somewhere between a third and a tenth of the width of a human hair, this was surely considered quite an achievement almost 50 years ago, even if it appears positively huge by today's standards.

Since then, feature sizes have tumbled, dropping to 1,000nm by 1985, and they have, more recently, reduced by a factor of around 1.4 to 1.5 per generation, which equates to a halving in terms of area. By and large, today's chips have a 14nm feature size, and the technology used to achieve that is referred to as the 14nm process.

While a reduction in feature size might have paralleled the increase in transistor counts over the decades, it might not be immediately obvious that there's a link between the two trends. At one time, decreases in feature size fuelled the increase in the clock frequency of a processor. So while the 4004 had a clock speed of 740kHz, by 1985 – when the feature size hit 1,000nm – the highest clock frequency achieved by the 80386 was 33MHz. By 2004, the 90nm process allowed the Pentium 4 to be clocked at 3.8GHz.

In the intervening 14 years, clock speeds have barely increased beyond this figure; indeed many of today's best-sellers actually have lower clock speeds. Even esoteric chips produced by binning – selecting those devices that tests show are capable of exceeding their design speed – can manage just 4.4GHz, with the option to boost a single core to 5GHz, temperature and power consumption permitting. Had speeds continued their previous trend, today's best processor would be clocked at 100GHz but, of course, escalating power consumption derailed this particular trend.

While clock speeds might have levelled out, feature sizes have continued to fall, for one very important reason: it's the driver of Moore's Law. An example will help shed some light on this. The Intel 4004 had 2,300 transistors and the chip measured 3 x 4mm. Had the feature size not reduced from 10,000nm, a high-end desktop chip such as the forthcoming Intel Core i9, with its estimated 7 billion transistors, would measure around 5m by 7m.

Admittedly, it's rather extreme to consider a current processor built using a 47-year-old process, even if it does lead to the bizarre picture of a chip that wouldn't come close to fitting in a desktop PC, let alone a handheld device. However, miniaturisation is about

QUICK GUIDE
The other Moore's Laws

Moore's Law refers to the number of transistors that can be crammed onto a chip, but lots of other trends show a similar exponential increase, and similarly the occasional decrease.

Comparing the speed of processors is tricky but memory chips are easy to quantify. Intel's first ever product, the 3101 RAM chip, had a capacity of 64 bits back in 1969, while today's largest chips have a capacity of 16 Gbits. That equates to 28 doublings, which works out at a doubling every 1.75 years. Turning to an exponential decrease, we only have to look at the price per transistor. The 2,300-transistor 4004 cost $200, which is a touch under 10 cents each, in 1971.

Let's stick with dollar pricing so we don't get embroiled in exchange-rate fluctuations, and pick a product costing much the same today. If we assume the Intel Core i5-8400 has 2 billion transistors, that works out at 0.00001 cents each. When we remove the effects of inflation to give a real terms comparison, the price has halved every 2.75 years.

QUICK TIP
The ultimate limit

However clever semiconductor manufacturers become, common sense tells us that transistor can't shrink forever. Today's 14nm chips are as small as 70 silicon atoms wide, so clearly the writing is on the wall.

far more than making small computers; it's a matter of economics. When a new processor is released, it might carry a price premium initially, but the market expects that it will eventually fall to about the price of its predecessor. However, without shrinking the feature size, doubling the number of transistors would mean that only half as many chips could be accommodated on a silicon wafer and this could only be compensated for by a massive price rise.

Another somewhat extreme example will serve to illustrate the enormity of the situation. It might be a gross over-simplification, but let's assume that the cost of making a processor is the fixed cost of fabricating a wafer divided by the number of dies (that is, chips) on the wafer. A photo of an unspecified Intel 7th generation processor wafer, based on the 14nm process, appears to contain roughly 500 dies. If the previous 22nm process had been used, there would only be half as many chips on the wafer, which implies that they'd have to cost twice as much. And if we go back to the 1997 250nm process – the oldest that would permit even a single 7th generation die to fit on a 300mm wafer – prices would be about 500 times greater than those of today's chips.

Shrinking in doubt

A clear link between the feature size and Moore's Law's continual doubling in the transistor count might be obvious, but where do we stand today? For a number of years, Intel promoted its 'tick-tock' model, which described the company's development philosophy. Ticks had a new feature size, and they alternated, on a yearly basis, with tocks, which introduced a new architecture.

This model was first announced in 2007 with the introduction of the 65nm process, and we saw four ticks and four tocks over seven years. In 2014, though, it was amended to a three-stage tick-tock-refresh model to eke out the 22nm process until the delayed 14nm process was ready to go. More recently, the company has abandoned both models; since the introduction of the 14nm process, we've seen one new architecture and two refreshes of that architecture.

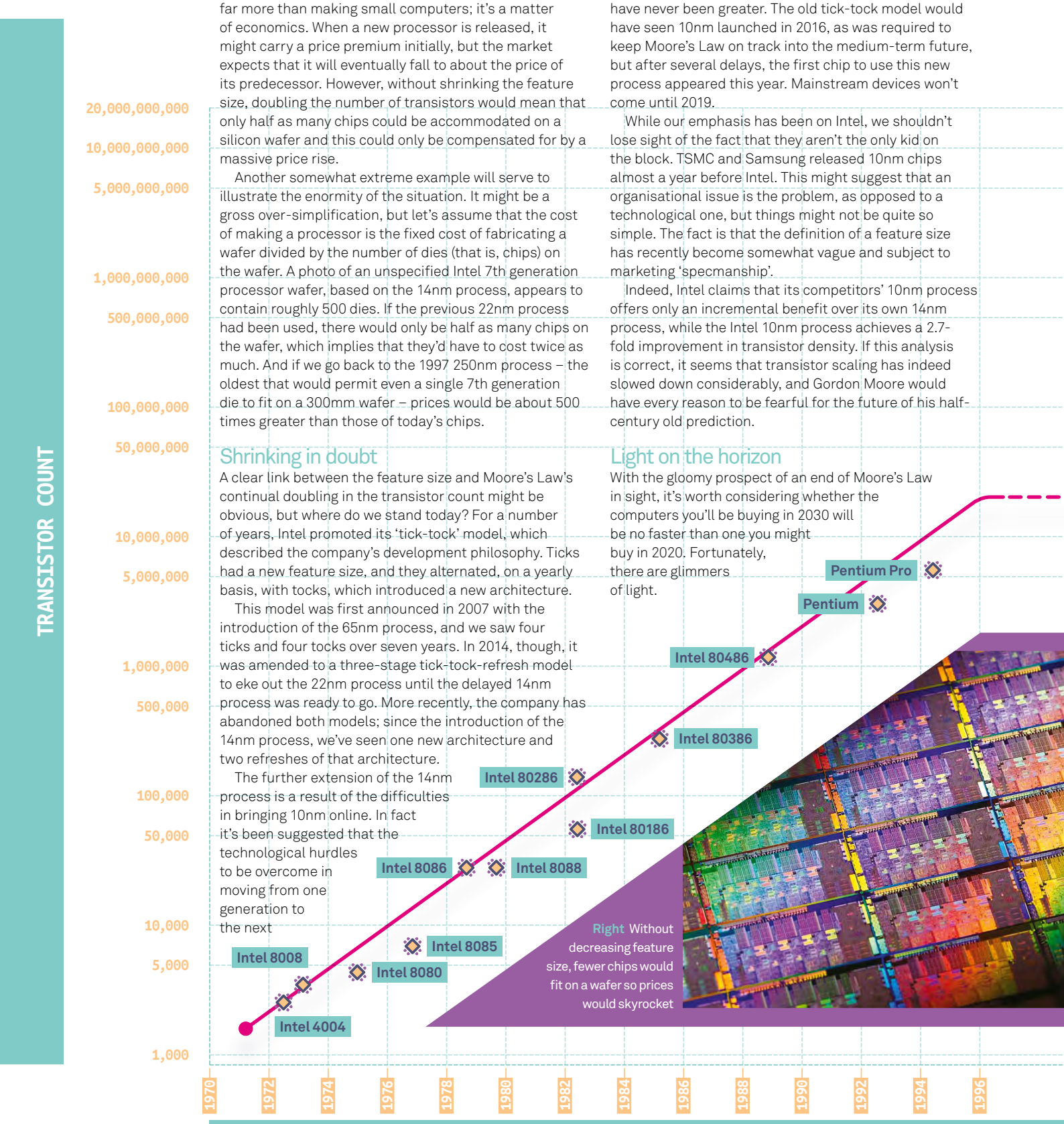
The further extension of the 14nm process is a result of the difficulties in bringing 10nm online. In fact it's been suggested that the technological hurdles to be overcome in moving from one generation to the next have never been greater. The old tick-tock model would have seen 10nm launched in 2016, as was required to keep Moore's Law on track into the medium-term future, but after several delays, the first chip to use this new process appeared this year. Mainstream devices won't come until 2019.

While our emphasis has been on Intel, we shouldn't lose sight of the fact that they aren't the only kid on the block. TSMC and Samsung released 10nm chips almost a year before Intel. This might suggest that an organisational issue is the problem, as opposed to a technological one, but things might not be quite so simple. The fact is that the definition of a feature size has recently become somewhat vague and subject to marketing 'specmanship'.

Indeed, Intel claims that its competitors' 10nm process offers only an incremental benefit over its own 14nm process, while the Intel 10nm process achieves a 2.7-fold improvement in transistor density. If this analysis is correct, it seems that transistor scaling has indeed slowed down considerably, and Gordon Moore would have every reason to be fearful for the future of his half-century old prediction.

Light on the horizon

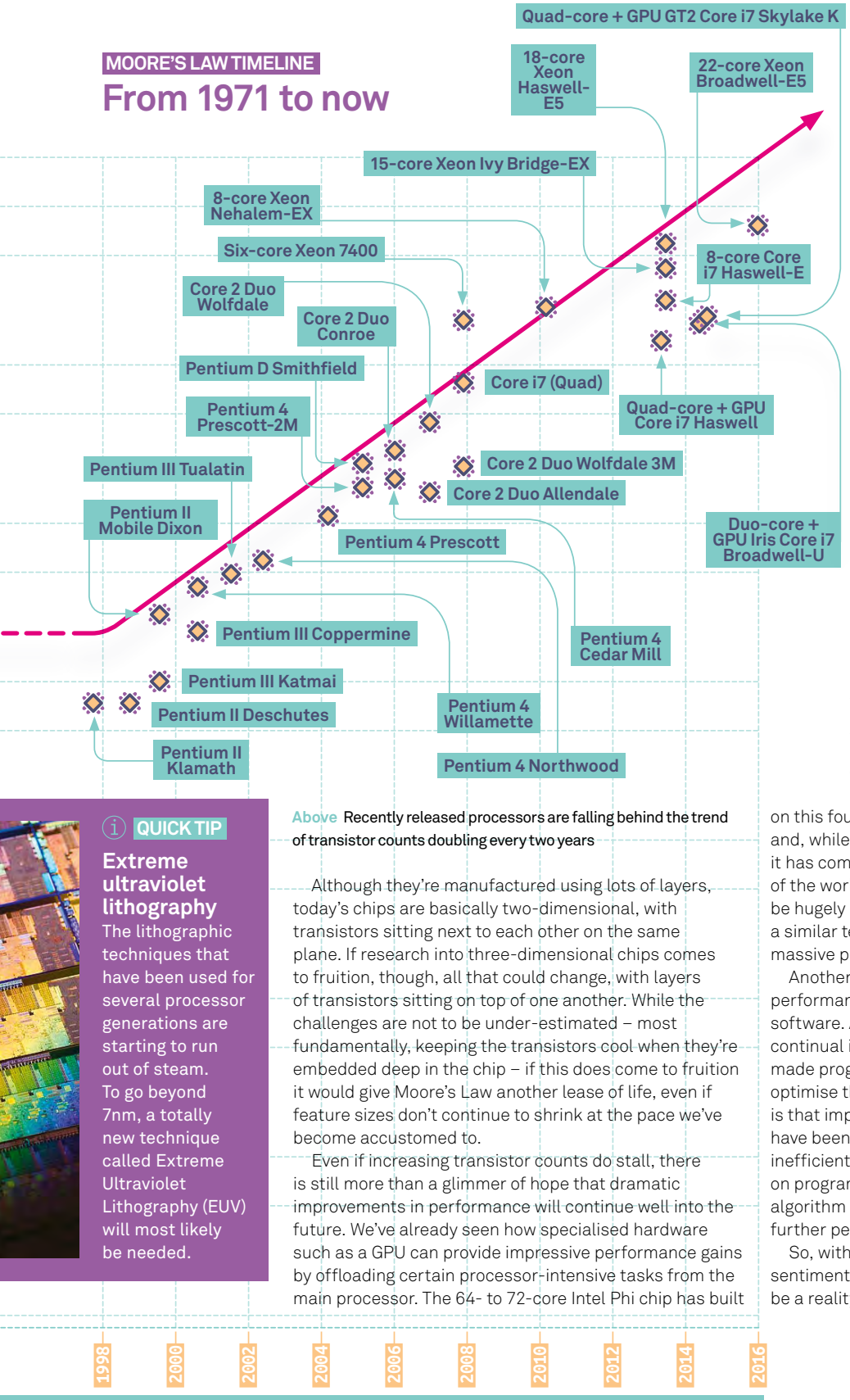
With the gloomy prospect of an end of Moore's Law in sight, it's worth considering whether the computers you'll be buying in 2030 will be no faster than one you might buy in 2020. Fortunately, there are glimmers of light.



Right Without decreasing feature size, fewer chips would fit on a wafer so prices would skyrocket

MOORE'S LAW TIMELINE

From 1971 to now



Above Recently released processors are falling behind the trend of transistor counts doubling every two years

Although they're manufactured using lots of layers, today's chips are basically two-dimensional, with transistors sitting next to each other on the same plane. If research into three-dimensional chips comes to fruition, though, all that could change, with layers of transistors sitting on top of one another. While the challenges are not to be under-estimated – most fundamentally, keeping the transistors cool when they're embedded deep in the chip – if this does come to fruition it would give Moore's Law another lease of life, even if feature sizes don't continue to shrink at the pace we've become accustomed to.

Even if increasing transistor counts do stall, there is still more than a glimmer of hope that dramatic improvements in performance will continue well into the future. We've already seen how specialised hardware such as a GPU can provide impressive performance gains by offloading certain processor-intensive tasks from the main processor. The 64- to 72-core Intel Phi chip has built on this foundation for high-performance applications and, while it can be used as a processor in its own right, it has commonly been used as a co-processor in some of the world's fastest supercomputers. The Phi might be hugely expensive but pundits are suggesting that a similar technology might give run-of-the-mill PCs a massive performance boost in a post-Moore's Law world.

Another key element in providing future gains to performance is a shift in emphasis from hardware to software. A commonly expressed sentiment is that continual increases in hardware performance have made programmers lazy, and removed the need to fully optimise their code. The result, it's been suggested, is that improvements in processor performance have been wiped out, at least in part, by increasingly inefficient programming. Perhaps a renewed emphasis on programming language design, the art of coding and algorithm efficiency could reverse this trend and provide further performance gains for years to come.

So, with a bit of luck and a following wind, perhaps the sentiment that 'This is as good as it gets' might not soon be a reality, after all.

QUICK GUIDE

Before Moore

Before microprocessors computers were built from small integrated circuits. Before that, individual transistors were wired together, and the very first stored-program computers used valves. The trend from 1971 onwards also holds true for all these preceding technologies, including famous names such as the Colossus and the Manchester Baby. Even the electro-mechanical Bell Calculator Model I, launched in 1940 and using relays, fits the trend. Only when we go back to the earlier purely mechanical calculators, do we find a slower but still exponential growth.

Copyright Intel

QUICK TIP

Extreme ultraviolet lithography

The lithographic techniques that have been used for several processor generations are starting to run out of steam. To go beyond 7nm, a totally new technique called Extreme Ultraviolet Lithography (EUV) will most likely be needed.

MAKE YOUR OWN PROJECTS WITH THE RASPBERRY PI

Learn the electronics, computing and coding skills you need to make your own projects with the Raspberry Pi, and let your imagination run wild

ON SALE NOW

FUTURE

Ordering is easy. Go online at:

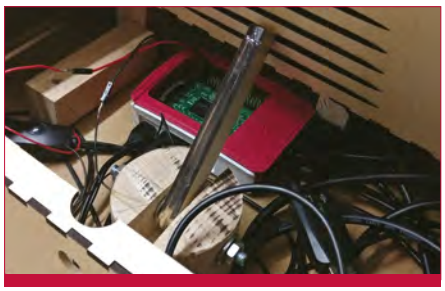
www.myfavouritemagazines.co.uk

Or get it from selected supermarkets & newsagents

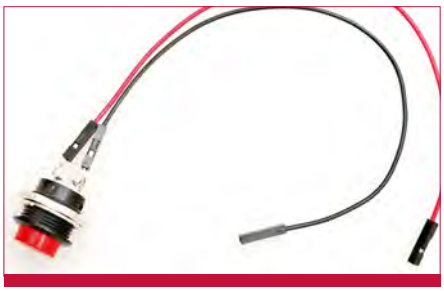
PRACTICAL Raspberry Pi

70 "Just go with the flow and solve the problems..."

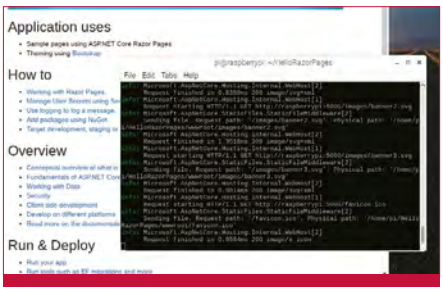
Contents



70 A Pi-controlled piano that guesses what you're playing



72 Build an image auto-uploader for the Raspbery Pi



76 Set up and Run .NET Core web apps on the Pi

MidiIdentifier

An elegant MIDI piano takes on an extra dimension with the ability to guess the music you're attempting to play

Andreas Scholz

Andreas studied computational linguistics for his Bachelors degree and is currently working on his Masters thesis.

Karsten Schick

Kartsten also studied computational linguistics and works as a computer scientist at Sulzer GmbH in Munich.

Like it?

You can contact them (reply not guaranteed!) on **andreas.scholz@fastmail.com** and **karsten.schick@gmail.com**.

Further reading

Head to **https://www.instructables.com/id/MidiIdentifier.**

The two German makers behind the MidiIdentifier are an interesting but unassuming pair that seem largely unaware of how good their little tinkering project is.

The MidiIdentifier seems to be the first that they have publicised, which is surprising given the quality of the work; it was largely built out of a love of the old Apple II computer design and the desire to do something clever but simple with music recognition.

This is such a stylish project – what inspired you to build it?

Thanks! We are both computer scientists. This sometimes makes it difficult for our family and friends to understand what we are actually doing. With this project, we wanted to build something that can be easily grasped (in a literal sense) and that we could show to our friends as well. It was also important for us to create something others could interact with in a fun and creative way, and we feel that music enables them to do just that.

> "We experimented with different algorithms to get as best a matching as possible while still being as forgiving to faulty input"

What made you decide to use the Raspberry Pi?

We didn't really think about it – it seemed the obvious choice at that moment. We knew we needed a computer that could be connected to a piano, a touchscreen and an external soundcard, as well as run the code. Together with its small-form factor, this made the Raspberry Pi the obvious choice for us.

What coding or libraries were involved in the project?

Our coding was done mainly in Python 3 along with a number of shell scripts. We made use of a variety of Python libraries including pygame for audio processing, tkinter for the user interface, mido to deal with the MIDI file format and the fluidsynth synthesizer for output.

How does the music identification aspect of the project actually work?

The MIDI format is actually quite complex – too complex for our purposes. So for our first step, we implemented a mechanism to convert the MIDI input from the piano – as well as all the MIDI files in our library – into a simpler format. On the basis of this format, we experimented with different algorithms to get as best a matching as possible while still being as forgiving to faulty input (wrong notes, octave and timing and so on) as possible. This was actually a very fun part of the project. It enabled us to be creative with algorithms and laugh in disbelief at how badly some of them worked.

Can you explain how you made the case?

We drew our inspiration for the case from the Apple II computer. After some paper prototypes and a very amateurish technical drawing, we used Adobe Illustrator to digitise the sketches. To generate the joints for the individual parts, we used a wonderful online tool called Joinery (**https://clementzheng.github.io/joinery**). Finally, we used a laser cutter to cut 3mm plywood and put everything together.

Would you do anything differently now? What advice can you give to someone attempting your project?

We would probably start developing on Raspbian right from the start. Initially, we developed the Python code on Windows, expecting for it to work the same way on a Linux system due to the generally very good cross-platform compatibility of Python. However, it turned out that Raspbian and Windows differ fundamentally in how they deal with MIDI sounds. That took a lot of time to figure out. As for recreating our project, don't be afraid of diverging from our write-up. In the end, there will always be something you encounter that we didn't. Just go with the flow and solve the problems.

What was the most challenging part of the project?

Getting MIDI files as well as individual notes to play correctly on the Raspberry Pi via the external soundcard. This sounds easy enough, but Raspbian did put some obstacles in our way there. The different behaviour on Windows and Linux complicated the matter even more.

What do you have in mind for your next project? Another musical project?

[KS] I don't have anything particular in mind, but I am interested in home automation and consider pursuing a project in that area.

[AS] Not a musical project, but I have an idea for an interactive art project I've been wanting to build for some time now. Stay tuned.

Components list

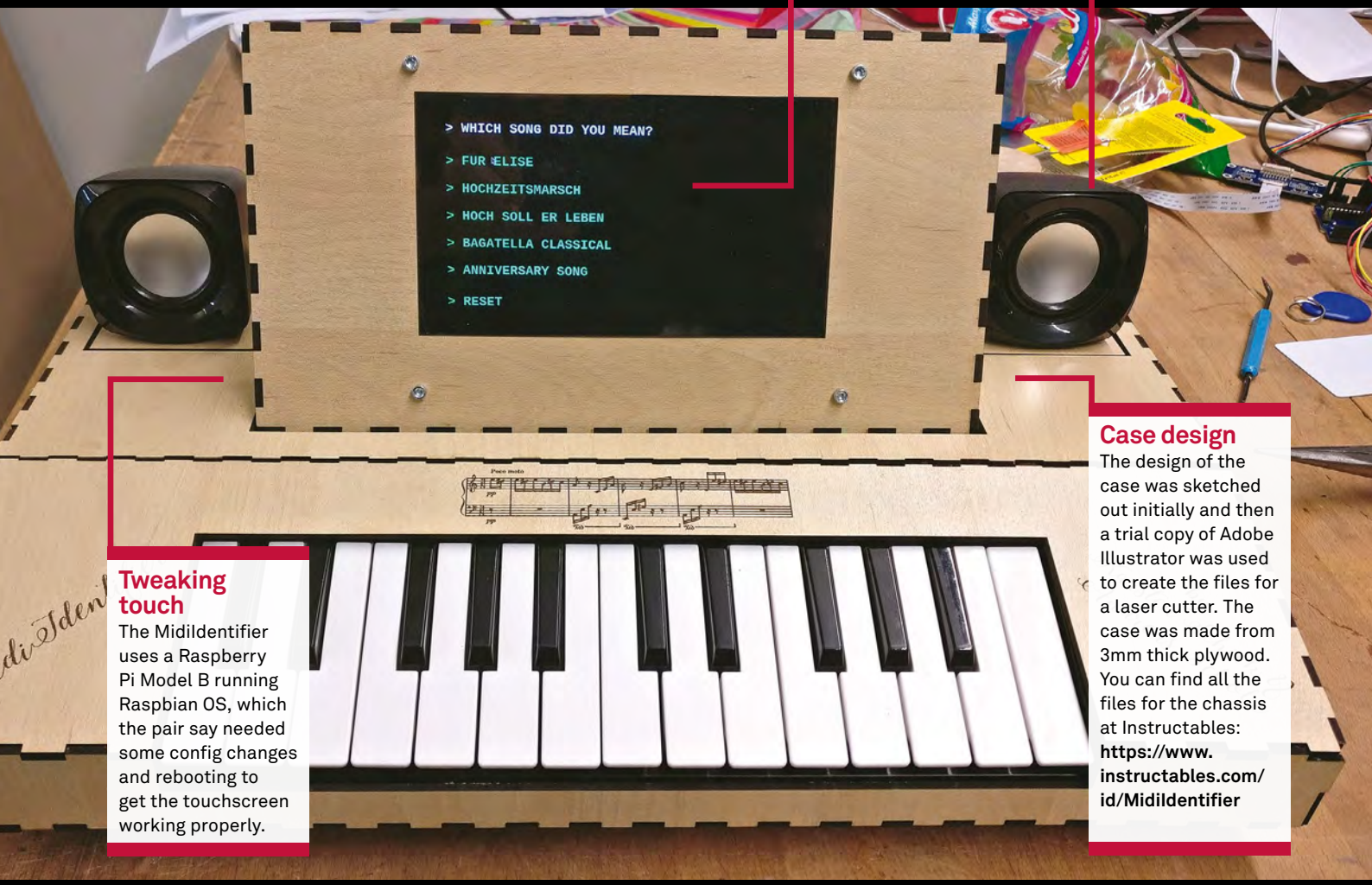
- Raspberry Pi 3 Model B (other models can be used)
- Capacitive touch-display 7-inch (Waveshare Raspberry Pi 7 Touch Screen LCD)
- Loudspeakers (Basetech Mini USB PC Speaker)
- MIDI USB keyboard (AKAI LPK25, 25-key ultra-portable USB MIDI keyboard controller for laptops)
- Wood for laser cutting (approx 3mm thick)

Adjustable screen

A wooden block was added inside the case to help hold up the screen in its horizontal position, and a small beam was attached to the block to adjust the screen viewing angles.

MIDI music

The speakers sit on plates that were hot-glued to wooden supports underneath, and the speakers were attached with double-sided tape.



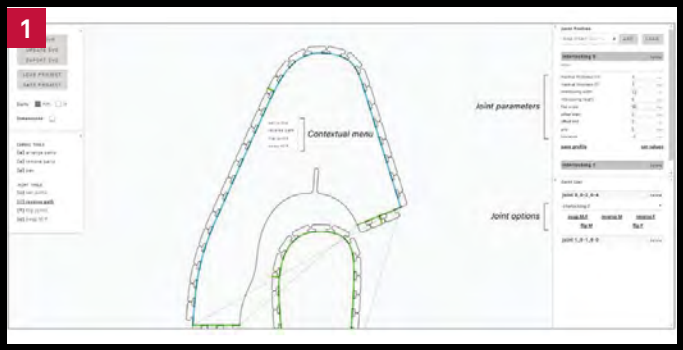
Tweaking touch

The Midildentifier uses a Raspberry Pi Model B running Raspbian OS, which the pair say needed some config changes and rebooting to get the touchscreen working properly.

Case design

The design of the case was sketched out initially and then a trial copy of Adobe Illustrator was used to create the files for a laser cutter. The case was made from 3mm thick plywood. You can find all the files for the chassis at Instructables: **https://www.instructables.com/id/Midildentifier**

1



2

Joining in with construction

Andreas and Karsten both credit the browser-based Joinery tool, created by Clement Zheng, as being a great help in designing and fabricating their laser-cut assemblies. This clever system takes a designer's defined connections between part edges and generates the joints that are required.

As long as it works…

Compared to the neatness of the case, the innards of the Midildentifer are a little on the scrappy side, but the pair say there wasn't an easy way to fit everything in and involved some "pushing, pulling, rasping, measuring, cutting, glueing and wiggling." The piano was inserted from the back and kept in place with wood.

Raspberry Pi: Build an image auto-uploader

Gain access to your images anywhere by combining the Pi Camera, Dropbox and Python code

Dan Aldred

Raspberry Pi enthusiast, teacher and coder who enjoys creating new projects and hacks to inspire others to start learning. Still trying to get a Kinect 360, Python and Pi to work together...

Resources

Raspberry Pi Zero W
https://www.raspberrypi.org/products/raspberry-pi-zero-w

Raspberry Pi Camera Module v2
www.raspberrypi.org/products/camera-module-v2

The Pi Camera Module v2 is a small 8Mp camera board that attaches to your Raspberry Pi. Combine this with the new Pi Zero W board – which has onboard Wi-Fi – a button and the official Raspberry Pi case, and you can build a simple 'point-and-shoot' camera.

The issue is, how do access your images? You could plug a monitor into the Pi, use a mouse and keyboard and find the folder with your images in. Or you could SSH in and transfer the images. But a better solution would be to automatically upload each image to your online Dropbox account, so that you can access your images anywhere and share them with your friends and family.

In this tutorial we will set up and configure a Pi Camera. Then we'll write a simple program to test that the Camera is functioning correctly. Next, we'll walk you through creating a Dropbox App and setting up the authentication token which enables you to access your account using Python code.

In the final section, we'll combine the PiCamera and Dropbox programs with GPIO Zero to facilitate image capture and upload via the touch of a button.

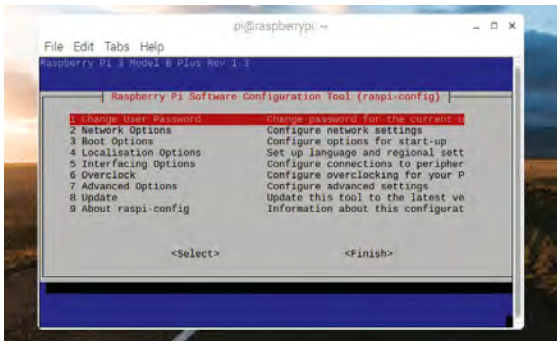
Above You could put the Pi in a smart camera-ready case

01 Setting up the Camera Module

Depending on where you purchased your Camera Module from, this first part may have already be done. Begin by connecting the ribbon cable to the Camera. Pull the small plastic clasp downwards and slide the ribbon in, ensuring that the blue strip is facing you. Push the clasp up, securing the ribbon into place.

Next, locate the Pi Camera socket. You will notice that it also has a clasp similar in style to the clasp on the Pi Camera. Pull this up to open it.

Take the ribbon and place it into the socket, with the blue strip facing towards the audio jack and away from the HDMI socket. Gently push the clasp back down to secure the ribbon into place. Make sure everything is firmly connected.



02 Enabling the Camera Module

The official Raspberry Pi OS comes preinstalled with the Camera Module software, with a Python package for interfacing with the camera. However, in the future you may need to reinstall or update the package when newer versions are released. Add the power supply to the Pi and then open the LXTerminal. Type **sudo apt-get update** then **sudo apt-get install python-picamera python3-picamera**. You can upgrade next time using **sudo apt-get update** and then **sudo apt-get upgrade**. By default, the Pi Camera is turned off, which of course means you can't use it. In the LXTerminal window type **sudo raspi-config** to open the configuration tool. Select choice 5, 'Interfacing Options', and enable the camera. Use the arrow keys to select exit and restart your Raspberry Pi.

03 Testing the Camera

Next, use the simple test program that displays the camera's view on your monitor for 10 seconds. If you can see the preview, it confirms that the hardware and software are working correctly. If not, check that the cable is attached correctly to the camera and Raspberry Pi. Also check that the camera is enabled in the configuration settings. Open your Python 3 editor and enter the program below. Then press **F5** to save and execute the program.

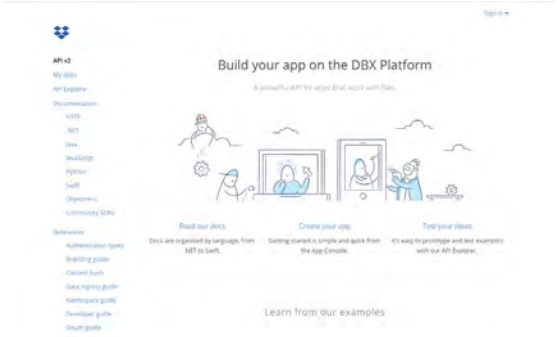
```
from picamera import PiCamera
from time import sleep
camera = PiCamera()
camera.start_preview()
sleep(10)
camera.stop_preview()
```

04 Sign up to Dropbox

If you already have a Dropbox account, you can skip this step. Otherwise open the web browser on the Pi and go to **www.dropbox.com**. Select the 'Create an account' option and then follow the steps to set up your Dropbox account. If you prefer, you can sign up using your Google account details.

05 Creating an app, part 1

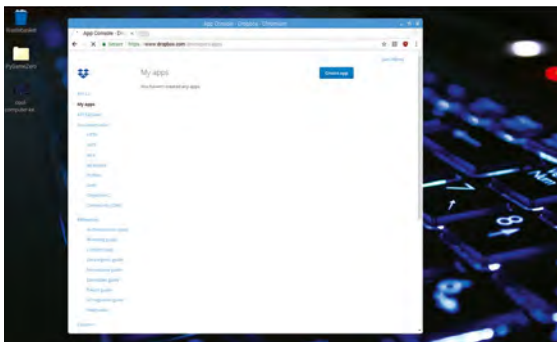
Once you have a Dropbox account go to **www.dropbox.com/developers/apps**. You will be presented



with a number of options down the left side of the page. The first entry references the API version number, then there's an entry called 'My Apps'. Select this option to be taken to your Apps page. If you have not previously created an App, this page will be blank. Select and click the blue 'Create app' button which appears at the top right of the page.

06 Creating an app, part 2

On this page you will be presented with further options. First select the 'Dropbox API' option which is free for personal use. Next you need to select the type of access that the app requires. For this project select 'App folder' access. This gives access to a folder on your Raspberry Pi which you will use to store the images you take with the Camera. The final part is naming the app, so call it something like 'image upload'. Finally, you'll have to agree to the Dropbox terms and conditions of use.



Portable 'Wi-Fi'

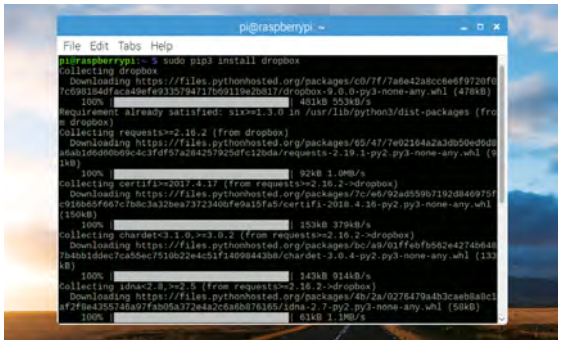
Enable the 'hotspot' capability on your mobile phone or tablet and tether your Raspberry Pi to it. This will enable you to take as many photos as you like and upload them in realtime – data constraints willing.

07 Creating an app, part 3

On the next page you will be presented with several configuration settings for your app. On the top line keep the status of the app set to Development, and 'Development users' to 'Only you'. You can also edit the app folder name to match the name you used in the previous step. Below these you'll find your app key and app secret, although for this project you do not require these. Instead, locate the 'Generated access token' towards the bottom of the page and if not present, click the button to generate one. This token is important and you will need to use it in your Python program, so make a note of it.

08 Install Dropbox for Python

Now that you have an authenticated Dropbox app, the next step is to install the Python Dropbox library



to enable access to your app and online folder from the Python program. Open the LX terminal and type `sudo pip3 install dropbox` to install the required software. At some stage in the future Dropbox will probably upgrade its software and release new features. To upgrade the software at a later date simply type `sudo pip3 install dropbox -upgrade` to ensure that the newest update is installed.

09 Upload an image, part 1

In this step, we'll test that the app configuration is correct by uploading an image file to Dropbox. First locate and save a suitable image to your Pi; in this tutorial the file is named `lud.jpg`. Open your Python 3 editor and type in the simple program below. We begin by importing the `dropbox` module, line one, and then assign your Dropbox authentication code to the `token` variable, line two. Enable authorisation on line three.

```
import dropbox
token = "xxxxxx"
dbx = dropbox.Dropbox(token)
```

10 Upload an image, part 2

Next we open the image file, line one. Note that the full file address is used. `rb` indicates that we want to read the file as binary, since it contains no text strings or characters. On line two, we read the contents of the open file and upload it. The `/lud.jpg` on this line is the

file name that will appear in the Dropbox folder. The option `mute = True` results in the user not receiving a notification of the upload. Lastly, add a quick `print` statement to confirm the upload. Press **F5** to save and run the program. Check your Dropbox account and the image will be stored there.

```
with open("/home/pi/lud.jpg", "rb") as f:
    dbx.files_upload(f.read(), '/lud.jpg', mute
= True)
print ("uploaded")
```

11 Live image capture and upload

As with the test program in the previous steps, begin by importing the `dropbox` module and then the `time` module so that you can add a short pause to the program code. Import `PiCamera` and `pause` to enable the program to loop. On line four, we import `Button` and then set up a global variable named `Image_Number`. This is used to store the image number, and on the next line we initially set this file number to zero. Finally, you'll need to assign the button to GPIO 2 and then set the `PiCamera` object to `camera`.

```
import dropbox
from time import sleep
from picamera import PiCamera
from signal import pause
from gpiozero import Button
global Image_Number
Image_Number = 0
button = Button(2)
camera = PiCamera()
```

12 Create a function

The function controls the image capture and upload. Start by naming the function, line one, and then add the global `Image_Number` variable, line two. On line three, increment the variable by one; this ensures that the file name for each image will be unique and will upload without overwriting the previous files. Combine the string `image` with the `Image_Number` value and `.jpg` to create the file name for each image taken. This results in files being named image1.jpg, image2.jpg, image3.jpg and so on. The final line adds the file name to the `pic` variable which is used later to upload the image file.

```
def Capture_and_Upload():
    global Image_Number
    Image_Number = Image_Number + 1

    camera.capture("image" + str(Image_
```

```
Number) + ".jpg")
    pic = ("image" + str(Image_Number) +
".jpg")
```

13 Access Dropbox

As with the test program in **Step 9**, we need to access Dropbox via your token, which is added on line two (replacing `xxxxxx`) and used on line three.

On line one, insert a `try` and `except` feature; this stops the program crashing if, say, the Pi is not online and cannot upload the file. On line four, a quick message lets you know that the program is ready to upload the image file. The final line opens the file location and then the image file, as binary data. Note that this uses the `pic` variable from the previous step, which contains our constructed file name.

```
try:
        token = "xxxxxx"
        dbx = dropbox.Dropbox(token)
        print ("ready")
        with open("/home/pi/" + pic, "rb")
as f:
```

14 Upload the image file

Now that the image file has been read it is ready for uploading to Dropbox. Use the function `dbx.files_upload()` and add the image file, followed by the file name. If you want a notification each time a file is uploaded, change `mute = True` to `False`. After the upload notify the user, line two, and then close the image file, line three. The delay is optional but works well for eliminating accidental double-presses of the button.

```
            dbx.files_upload(f.read(), '/
image%s.jpg' %(Image_Number), mute = True)
             print ("uploaded")
             f.close()
             sleep(5)
```

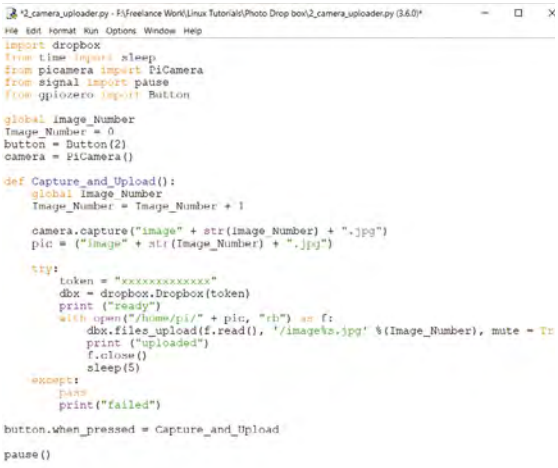
15 Add the except

On the first line add the `except` condition that runs if it is not possible to upload the image file to Dropbox. The program passes the step, line two, and then prints a message to let you know that the upload has failed, line three. The image is still stored on your Pi, so you can access it later.

Line four uses the GPIO Zero function `button.when_pressed` to check for the button being pressed. If it is, then the function created in Step 12 `Capture_and_Upload()` is run. Add the final line `pause()` which ensures that the program loops and waits again for the button to be pressed. Save your program code.

```
    except:
        pass
        print("failed")

button.when_pressed = Capture_and_Upload
pause()
```



16 Adding the button

The final part of the setup is to add a physical button which triggers the program to take an image and upload it. If you don't have a button you can still use jumper wires by touching them together. Attach the first wire to pin GPIO 02 and the other pin to one of the GND pins. This creates the circuit.

Now run the program and press the button to take the picture. If Dropbox already contains a file with the same name, the program will not upload the file and you will get the message "failed". We'll leave it up to you to tidy up the error-handling and maybe add in some new functions to make the whole thing a bit friendlier.

Camera effects

The Camera Module software is nifty and can be used to manipulate the image. You could add Instagram-style filters to your pictures before you upload them. In the main program (**Step 12**), locate the line with `camera.capture()`. On the previous line add the following code: `camera.image_effect = 'emboss'`. This will add a simple emboss filter to the image. There are several to choose from and experiment with. How about 'watercolor', 'oilpaint', 'hatch', 'pastel', 'film', 'blur', 'colorswap', 'washedout' or the classic 'cartoon'?

Tutorial files available:
filesilo.co.uk

Kevin Wittmer

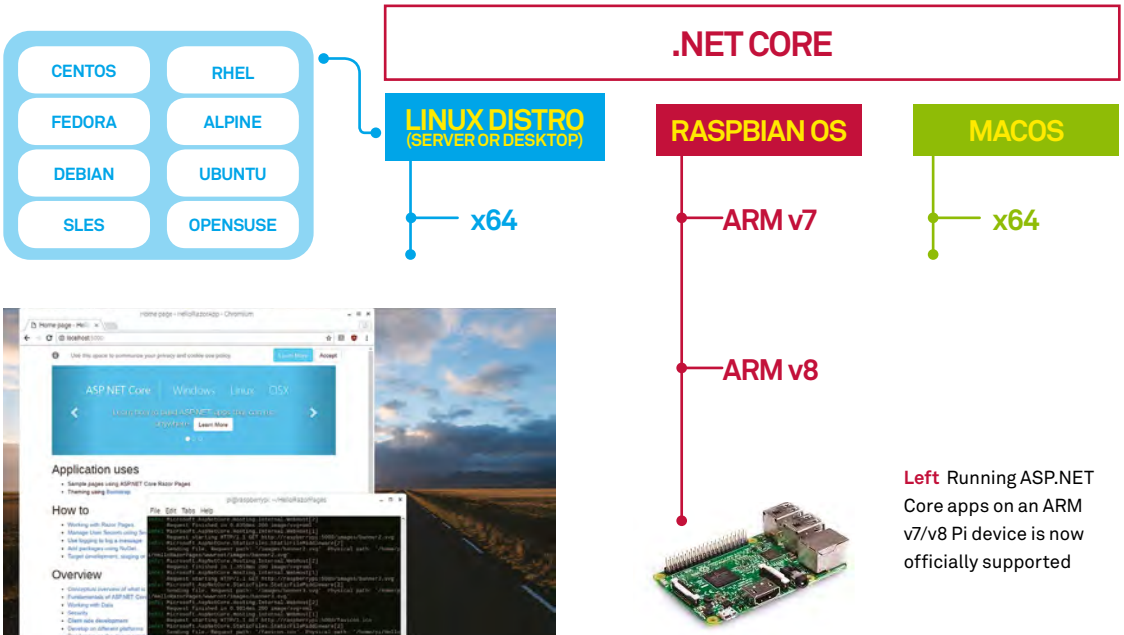
Kevin Wittmer lives in the Chicago area and works as a senior technologist and manager in the Bosch digital business unit. He enjoys most things Linux.

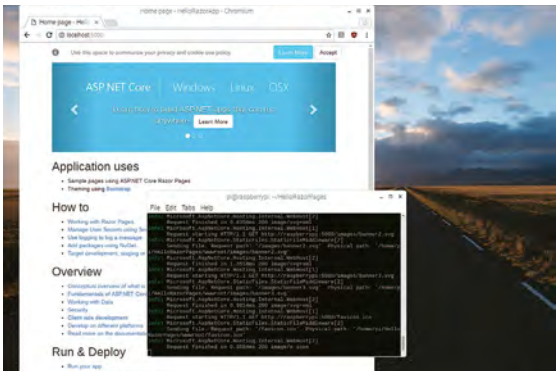
Resources

- **.NET Core Home** www.dot.net
- **ASP.NET Core** https://docs.microsoft.com
- **Orchard Core CMS** https://github.com/OrchardCMS/

Run .NET Core web apps on the Raspberry Pi

Microsoft officially extends Linux support to the Pi – find out how to use it





Left Running ASP.NET Core apps on an ARM v7/v8 Pi device is now officially supported

Above .Net Core expands its support beyond Linux and MacOS to now include Raspberry Pi.

.NET Core continues to broaden its reach, and with release 2.1 Arm chips are officially supported. This makes it possible to run .NET Core apps on Raspberry Pi devices. Release 2.1 further expands coverage of the .NET Core Runtime and SDK beyond CentOS, Red Hat Linux, Fedora, Debian, Ubuntu and SUSE. Now, in addition to Arm-based Pi environments, you can also deploy .NET Core apps to Alpine Linux, which is ideal for Docker scenarios.

With respect to the Pi, Armv7 or Armv8 chips are supported by .NET Core – however, Pi Zero is not, nor are other Armv6 derivatives. The experience of running .NET Core apps on a Raspberry Pi is reasonably good with the 2.1 release, and there are a few hints that we'll cover in this article to help improve it.

If you're just getting started with .NET Core, we'll first briefly highlight the recent history of .NET in the Linux space to help establish a basis and then introduce the steps to install .NET Core – first to your desktop or server environment, and then to your Pi device. To keep things simple, we'll use the web app templates provided in the .NET Core SDK and then conclude with highlighting the open-source CMS project Orchard running on Pi.

A new era of .NET

.NET running outside of a Windows context is not new; in the Linux context, Mono did it first. As the Mono open-source effort evolved, the capability to develop C# applications and programs increased as the supported API surface area slowly expanded. APIs that offered ASP.NET and WinForms capability in the Linux context were contributed back to the Mono open-source project. This effort continued slowly for several years; however, the supported API surface area of Mono often lagged behind new technology or APIs provided in the Windows-based .NET Framework.

Then, beginning in 2014, the .NET technology stack was made open source by Microsoft and a new effort to make the .NET technology stack run on multiple operating systems was launched. As development in .NET Core progressed across several public GitHub projects, community interest in running .NET on Linux

was slowly rekindled. June 2016 saw the release of .NET Core 1.0 which shipped with support for multiple Linux distros along with the capability to run from Linux-based containers. Now, more than two years later, .NET Core has climbed to version 2.1 and adds support for Raspberry Pi.

Install .NET Core SDK to Ubuntu

An easy way to begin developing .NET Core apps for Raspberry Pi is to work from your primary Linux environment. In this way, you simply use the standard tools available in the .NET Core SDK, and target ARM-based Pi devices during the .NET Core app publishing process. If your Ubuntu 18.04 environment does not already have the .NET Core tools installed, follow the command series below to add Microsoft repositories as a source for the local package manager (which is a necessary preliminary step), then proceed to install the .NET Core SDK:

```
wget -q https://packages.microsoft.com/config/ubuntu/18.04/packages-microsoft-prod.deb
sudo dpkg -i packages-microsoft-prod.deb
```

Following the commands above, the Debian package file **packages-microsoft-prod** will be prepared, unpacked and installed. After this, the next step is invoke the update command of **apt-get** to have these package sourcing changes become effective. Once updated, there is one additional preliminary step – install the HTTPS transport (dependency) to Ubuntu's Advanced Packaging Tool (APT). With this dependency installed, you are ready to install the .NET Core SDK. The command series shown below illustrates the entire sequence:

```
sudo apt-get update
sudo apt-get install apt-transport-https
sudo apt-get install dotnet-sdk-2.1
```

After issuing the commands above, you should expect a lengthy amount of package-orientated textual output dumped to the console, including package installation details for .NET Core and ASP.NET Core.

```
..
The following NEW packages will be installed:
  aspnetcore-runtime-2.1 dotnet-host dotnet-hostfxr-2.1 dotnet-runtime-2.1
  dotnet-runtime-deps-2.1 dotnet-sdk-2.1 liblttng-ust-ctl4 liblttng-ust0 liburcu6
..
```

With packages of the .NET Core SDK installed, type the command (below) to quickly verify your .NET Core install:

```
dotnet --info
```

Successful verification will result in the Runtime and SDK versioning details printed to the console. With the SDK installation complete, you are ready for command-line driven development. The **dotnet** command is the primary SDK tool to interact with. The **new** argument of **dotnet** supports template-based scaffold code generation and offers a number of out-of-the-box project templates to help launch initial code development. Project templates available include console, web UI, web API, unit test and project/solution. In this example we will use the MVC template project.

The MVC template project will generate the scaffolding code that's required, which includes models, views and controllers based on the ASP.NET Core programming model. To generate scaffolding for a new MVC project you will need to create a subdirectory and execute the commands (as we've shown below). Afterwards, use the **tree** command to make a quick inspection of what was copied over from the MVC template. The source tree output will show the base structure of the web app with placement folders for Controllers, Models and Views, like this:

```
mkdir HelloMvc; cd HelloMvc
dotnet new mvc
tree
├── appsettings.Development.json
├── appsettings.json
├── Controllers
│   └── HomeController.cs
├── HelloMvc.csproj
├── hello-mvc-tree.txt
├── Models
│   └── ErrorViewModel.cs
├── Program.cs
├── Properties
│   └── launchSettings.json
├── Startup.cs
├── Views
│   ├── Home
│   │   ├── About.cshtml
│   │   ├── Contact.cshtml
│   │   ├── Index.cshtml
│   │   └── Privacy.cshtml
│   ├── Shared
│   │   ├── _CookieConsentPartial.cshtml
│   │   ├── Error.cshtml
│   │   ├── _Layout.cshtml
│   │   └── _ValidationScriptsPartial.cshtml
│   ├── _ViewImports.cshtml
│   └── _ViewStart.cshtml
└── wwwroot
    ├── css
    ├── favicon.ico
    ├── images
    ├── js
    └── lib
```

Generation of the project template will indicate that the 'ASP.NET Core Web App (Model-View-Controller)' was created successfully. The **dotnet restore** command will be executed as a post-creation step. This step will

dotnet command and arguments

dotnet
The multi-purpose dotnet command line tool

dotnet new
Creates a new project, configuration file, or solution based on the specified template. Templates include console, classlib, xunit, mvc, razor, angular, react and webapi

dotnet build
Builds a project and all of its dependencies

dotnet clean
Cleans the output of a project

dotnet publish
This command compiles the application, reads through its dependencies specified in the project file, and publishes the resulting set of files to a directory. The output includes assemblies, dependencies, and config files

download and cache (to disk) the NuGet-based project dependencies specified in the main project file. Project template generation now also includes installation of a development certificate, which is a new behaviour added in the .NET Core SDK 2.1 release.

With the project scaffolding generated, you are ready to compile the C# source code and build project-related assemblies. To build a .NET Core project, use the **dotnet build** command. This will load the .csproj file and apply project related settings. C# source files located below the parent folder are recursively scanned and compiled as part of the compilation stage.

```
dotnet build
```

To run the project, simply type **dotnet run**, which will identify the entry point in the web app, execute the **Startup** class, and launch the Kestrel-based web server host process.

```
dotnet run
```

Deploy to Pi as a self-contained app

Working from within the scaffold code, you can easily create new C# classes and introduce methods that make up your app from your Visual Studio Code, vim or your favourite editor. As you write new code, associated unit tests that exercise app behaviour can be executed using the **dotnet test** command. Eventually, after several iterations in the inner (code-build-test-debug) loop, you will reach a point where you will be ready to deploy to the Pi itself.

> “With the project scaffolding generated, you are ready to compile the C# source code and build projects”

To target an ARM-based Raspberry Pi device, use the **dotnet publish** command. The **-r** argument of **dotnet publish** is used to specify a Linux ARM-based distribution. Executing the command **dotnet publish** results in the generation of a self-contained deployment which will include the .NET Core Runtime as part of the standalone distributable file set.

To prepare your project for ARM distribution and copy files of the self-contained deployment from your (Linux) host to Raspberry Pi, execute the following command sequence:

```
dotnet publish -c Release -r linux-arm
scp -r /home/kevin/HelloMvc/bin/Release/
netcoreapp2.1/linux-arm/publish/* pi@
raspberrypi:/home/pi/HelloMvcArmApp
```

The MSBuild engine will fire off, installing .NET-based NuGet library dependencies to be included in the standalone distribution. Note that shared object files such as **libcoreclr.so** are also included in this file copy operation.

Once the standalone binary assemblies have been copied to the target Pi device, you are ready to login to the Pi device and execute the MVC web app. After SSH-ing into your Pi, change directory to the destination directory previously specified. Then use **dotnet exec** to explicitly launch the parent assembly containing the Program object and providing an executable Main entry point for the .NET Core runtime. Invoking dotnet with the lesser-known **exec** argument uses the binary prepared in the self-contained distribute phase while also improving app launch performance, as the restore and build steps are not triggered.

```
dotnet exec HelloMvcArmApp.dll
```

Once the following output is printed to the console, you have the confirmation that you’re ready to fire up a web browser and go to **http://raspberrypi:5000**.

Versioning of .NET Core (SDK, Runtime)

Grasping versioning of .NET Core can be difficult. Why? Because the model for versioning has changed as the open source project has evolved. You must first recognise that .NET Core consists of two high-level components: .NET Core Runtime and the .NET Core SDK. These two entities follow a different versioning scheme, although the SDK versioning maintains an association with Runtime versioning. Semantic Versioning (SemVer) is applied to the .NET Core Runtime version, with three components: major, minor and patch.

The SDK, however, doesn’t follow this system, but in a new scheme pins the first two positions of the .NET Core SDK version to the .NET Core Runtime release. The third position of the SDK version number represents the minor and patch number, calculated using the simple formula: minor * 100 + patch version. For example, a minor version 1 with a patch version 7 will result in a patch version number of 107.

In practice, significant changes occurring to .NET Core are reflected in a change to the major, including breaking code changes. Minor changes carry less impact and often are associated with new APIs or behaviours. Note that each version of the SDK can create applications for the identified runtime (version) or any lower version.

As a general trend, the .NET Core SDK receives updates more frequently than the .NET Core Runtime and this version numbering scheme helps communicate relationships between these two major components. For further information, see **https://docs.microsoft.com** and search for .NET Core versioning.

Download and Install .NET Core onto Pi

Installing .NET Core SDK onto a bare-metal Pi involves first installing dependencies, specifically the package for stack exception unwinding. Also, the **gettext** package is needed as it’s used by the .NET Core SDK and Runtime to produce multi-lingual messages. Use the following command block to update the local package list and

specify the packages **libunwind8** and **gettext** for install.

```
sudo apt-get -y update
sudo apt-get -y install libunwind8 gettext
```

The script shown below will download, decompress, extract and move the .NET Core SDK to the **opt** subdirectory. As the .NET Core open source code base is being updated on a regular basis, you'll need to identify the most recent version of .NET Core to install. To assist in version management, the Bash script will read in the .NET Core version parameter and use this to download the target SDK version. Note that the command series below defaults to Arm32, although an Arm64-based SDK and Runtime is also available.

```
#!/bin/bash
# Example usage: ./dotnet-sdk-pi-install.sh 2.1.401
echo "Will download, expand and install version $1 of the .NET Core SDK."
sudo apt-get -y update
sudo apt-get -y install libunwind8
sudo apt-get -y install gettext
wget https://dotnetcli.blob.core.windows.net/dotnet/Sdk/$1/dotnet-sdk-$1-linux-arm.tar.gz
mkdir -p dotnet-installers/$1
mv dotnet-sdk-$1-linux-arm.tar.gz dotnet-installers/$1
sudo mkdir -p /opt/dotnet/$1
sudo tar -xvf dotnet-installers/$1/dotnet-sdk-$1-linux-arm.tar.gz -C /opt/dotnet/$1
sudo ln -s /opt/dotnet/$1/dotnet /usr/local/bin
dotnet --info
```

As part of the setup, this script will download the .NET Core version specified as input and include the specified version in the directory path structure. Soft-linking to **/usr/local/bin** then determines which version of .NET Core is available in the shell context – for example, which version of dotnet will be used when invoked with the **new** or **build** arguments. It's worth noting that as of writing Microsoft has not released Runtime or SDK packages for Raspbian OS, though this might change in the future.

Microsoft also has a similar script, dotnet-install script, to perform a non-admin installation of the .NET Core SDK, but this script hasn't been updated for Pi support (again, this could change in the future). Finally, note that the documentation maintained at the dot.net website sets the DOTNET_ROOT env variable to the top-level folder; in the example above, this would be **/opt/dotnet/<sdk-version>**.

Develop locally on Pi

In the example below, we'll use the Razor template project to create a newly scaffolded project directly from the Pi shell prompt. Razor is a page-based model programming model somewhat reminiscent of Active Server Pages for .NET (ASP.NET). In the ASP.NET Core context, Razor delivers the benefits of a web page-programming model while also providing access to MVC Core plumbing. This is possible because Razor Pages are built on top of ASP.NET Core MVC. Therefore, all the goodness from the ASP.NET Core MVC framework, including routing, ModelState, and so on, can be accessed by code placed into Razor pages.

However, Razor programming does not require recognition or understanding of MVC, and thus you are able to happily code away and only apply the PageModel programming constructs. To dive into Razor programming, visit the getting started with Razor Pages tutorial available at **https://docs.microsoft.com**.

```
dotnet new razor
tree
├── appsettings.Development.json
├── appsettings.json
├── HelloRazor.csproj
├── Pages
│   ├── About.cshtml
│   ├── About.cshtml.cs
│   ├── Contact.cshtml
│   ├── Contact.cshtml.cs
│   ├── Error.cshtml
│   ├── Error.cshtml.cs
│   ├── Index.cshtml
│   ├── Index.cshtml.cs
│   ├── Privacy.cshtml
│   ├── Privacy.cshtml.cs
│   ├── Shared
│   │   ├── _CookieConsentPartial.cshtml
│   │   ├── _Layout.cshtml
│   │   └── _ValidationScriptsPartial.cshtml
│   ├── _ViewImports.cshtml
│   └── _ViewStart.cshtml
├── Program.cs
├── Properties
│   └── launchSettings.json
├── Startup.cs
└── wwwroot
    ├── css
    ├── favicon.ico
    ├── images
    ├── js
    └── lib
```

The results of the tree shown above quickly illustrate the C#-centric .cshtml and.cs files of this web page programming model. You can now fire up your favourite programming editor and begin to modify and/or create new C#-based web pages (using the page dotnet new template). For simple demonstration purposes, we will only modify the source of Program.cs, specifically the hostname pattern passed into the **UseUrls()** method:

```
WebHost.CreateDefaultBuilder(...).UseUrls("http://*:5000");
```

To restore, build and run the Razor Web program from the Pi context, type the following. Note that depending

More dotnet command and arguments

dotnet run
Runs source code without any explicit compile or launch commands

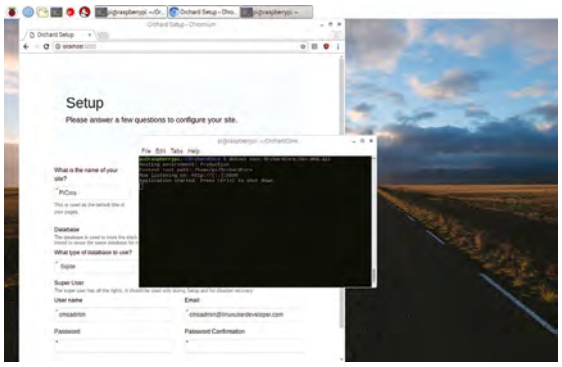
dotnet exec
An advanced parameter passed to the host that indicates only to execute the specified assemblied identified as a command-line argument

dotnet test
Inspect and execute any unit tests identified in the project file

dotnet pack
Builds the project and creates NuGet packages. The result of this command is a NuGet package

dotnet restore
Uses NuGet to restore dependencies as well as project-specific tools that are specified in the project file

dotnet help
Get help on the capabilities of dotnet tool and the associated arguments for the respective capabilities



Right The setup screen of the Orchard Content Management System. With just a few inputs and selections, you can host a CMS from your Raspberry Pi

on your model of Raspberry Pi, you might have to wait some time before the multi-step .NET Core process completes. This experience will likely highlight the benefits of building web apps on a desktop or server environment, publishing the binaries, and copying over the self-contained .NET web app to the Raspberry Pi via secure copy. This is especially beneficial in non-trivial examples such as the example that we'll look at next with the Orchard Core CMS.

```
dotnet run
```

With the following output printed to console, you'll have the confirmation that it is okay to open your browser and go to **http://raspberrypi:5000**.

```
Hosting environment: Production
Content root path: /home/pi/HelloRazorPages
Now listening on: http://[::]:5000
Application started. Press Ctrl+C to shut down.
```

Open-source CMS with .NET Core and Pi

Previous examples demonstrated the technique of creating, building and/or deploying scaffolding project code generated by invoking the MVC and Razor (pagemodel programming) templates that ship with the .NET Core SDK. From these examples you can quickly gain an impression of the subtle differences involved in targeting Raspberry Pi devices using the .NET Core SDK and .NET Core Runtime.

In this next example, we'll briefly highlight Orchard Core, a CMS system that is open-source and built 100 per cent on .NET technology. Recently, Orchard was redeveloped to take advantage of the new ASP.NET Core programming model, with the outcome being Orchard Core. This gives us a more advanced example that comes with real-world possibilities out of the box, and name content management capabilities.

Orchard Core is an open-source project using the BSD license and largely maintained by community members outside of Microsoft. It includes an extensibility model for both modules and themes, a dynamic content type system that makes heavy use of C# dynamics, supports customisation, localisation and more. The following command series will clone the GitHub open source project of Orchard Core, restore the NuGet-based dependencies and build the source project.

```
git clone https://github.com/OrchardCMS/OrchardCore.git
cd OrchardCore/
dotnet build
```

After successfully building Orchard Core from the host environment, you can quickly verify the run state by navigating to the **OrchardCore.Cms.Web** subdirectory and using the **dotnet** command to run the main assembly.

```
cd src/OrchardCore.Cms.Web/
dotnet run
```

After bringing up the home page on port 5000 (in our case, this is **http://ubuntusrvr18:5000**), you can create a new site, apply one of the built-in recipes and begin to build a simple, content-driven web site.

After validating, Orchard Core will successfully build and run from the local Linux environment. You can repeat the command series provided in the initial example to copy Arm-based published binaries to the Raspberry Pi environment. The following command series will accomplish this using the **dotnet publish** command to prepare the binary distribution for the Arm platform.

```
cd OrchardCore/src/OrchardCore.Cms.Web/
dotnet publish -c Release -r linux-arm
scp -r /home/kevin/OrchardCore/src/OrchardCore.Cms.Web/bin/Release/netcoreapp2.1/linux-arm/publish/* pi@raspberrypi:/home/pi/OrchardCore
```

As before, first SSH into the target Pi environment. Complete the additional setup steps for Orchard Core, including any additional database configuration (Orchard Core can be configured to work with multiple DB providers). Once the configuration is done, you are ready to fire up Orchard Core from the command line using the **exec** argument and specify the main assembly name, OrchardCore.Cms.Web.dll. Once the server process has finished initialising on port 5000, you can then proceed to access the admin console of Orchard Core and create a new CMS site.

```
ssh -l pi raspberrypi

cd OrchardCore
dotnet exec OrchardCore.Cms.Web.dll
```

After executing, point your web browser to **http://raspberrypi:5000** to access the Orchard Core (initial instance) setup page.

Key aspects to take away from this article include the benefits of self-contained apps as well as the unofficial **dotnet exec** command to help improve web app start-up performance. And with this last example, you're now armed with the knowledge to fire-up .NET Core on your Pi!

Reviews

81 **Group test** | 86 **Hardware** | 88 **Distro** | 90 **Free software**

GROUP TEST

Disk cloning tools

Managing a network of computers can be an involved process, so here are the best tools to help you image and roll out multiple machines with ease

Clonezilla

One of the oldest open source disk imaging and cloning tools, Clonezilla is available in multiple versions. It supports a wide variety of file systems, including many of the popular file systems for Linux, Windows and macOS. It's written primarily in Perl, and is released under GPL v2.

https://clonezilla.org

FOG Project

FOG is another popular tool for imaging disks, but unlike the other options here it uses a PHP-based web interface that you can access from any computer on the network. Besides cloning disks, you can also use FOG to manage installations and peripherals on the network.

https://fogproject.org

G4L

G4L – Ghost4Linux – does image and clone disks, but isn't a direct replacement for Symantec Ghost, not does it only work with Linux. Like most of the other options here, G4L is available as a bootable Live CD and offers various options to image several types of file systems.

http://bit.ly/lud_g4l

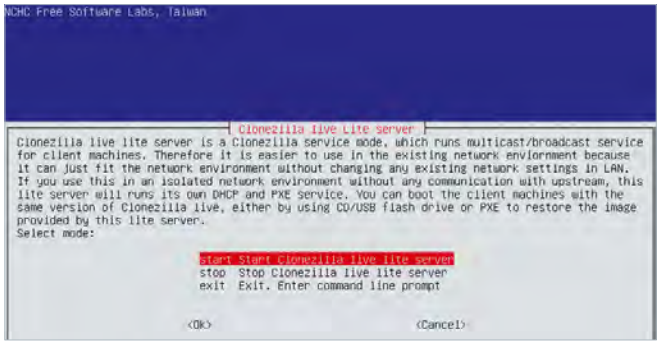
Redo Backup

Redo Backup & Recovery is one of the easiest to use bare-metal cloning and recovery tools. The small bootable Live distribution works across Linux and Windows file systems, and although it hasn't been updated in quite some time, it works flawlessly without unexpected errors.

http://redobackup.org

Clonezilla

A feature-rich tool that scales well in terms of reach and application


NCHC Free Software Labs, Taiwan
Clonezilla live lite server
Clonezilla live lite server is a Clonezilla service mode, which runs multicast/broadcast service for client machines. Therefore it is easier to use in the existing network environment because it can just fit the network environment without changing any existing network settings in LAN. If you use this in an isolated network environment without any communication with upstream, this lite server will runs its own DHCP and PXE service. You can boot the client machines with the same version of Clonezilla live, either by using CD/USB flash drive or PXE to restore the image provided by this lite server.
Select mode:
start Start Clonezilla live lite server
stop Stop Clonezilla live lite server
exit Exit, Enter command line prompt
<OK>
<Cancel>

Clonezilla can also clone partitions and disks on the fly, but we'd advise you to create and clone from images for safety

Usability

Clonezilla is one of the most recommended tools on Linux forums for imaging an old disk to a bigger, new disk thanks to its ncurses-based front-end, which is intuitive to navigate. The menus offers sensible defaults so you'll find yourself pressing ahead with the recommended options most of the time. Just make sure that the partition to be cloned isn't mounted.

Malleability

When cloning, you're shown a list of disks or partitions, depending on whether you want to clone complete disks or individual partitions. You also get options to compress the cloned images using either the gzip, bzip2 or lzo compression algorithms. You can keep them in a locally attached device, such as a USB disk attached to the computer, or on another computer via the network.

Network use

In addition to cloning single machines, you can also use Clonezilla for multicast cloning via two available options. Clonezilla Server Edition requires setting up a Diskless Remote Boot in Linux (DRBL) server to broadcast images across the network. On a smaller network, you can use the Clonezilla Lite Server mode on the bootable CD to set up a temporary server.

Restoration

Restoring individual partitions or complete disks from a clone isn't a complicated process. You do have to ensure that the destination partition is equal to or larger than the source, as it can't restore an image to smaller disks.

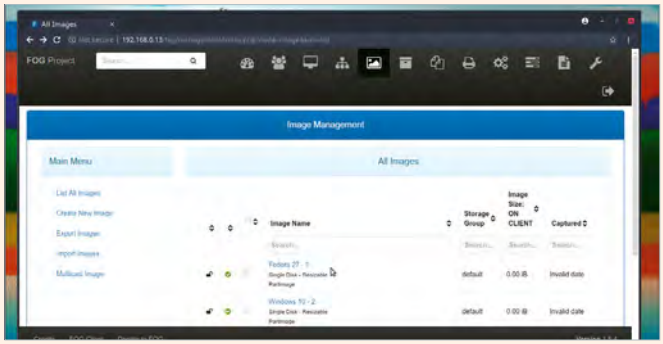
Overall

An oft-recommended tool for bare-metal cloning that works as advertised. It's well documented and can be used to effortlessly clone single or multiple computers with relative ease.

8

FOG Project

The only tool with a web-based dashboard to manage the process



Unlike the other solutions, you can image and clone computers by accessing the FOG dashboard from a mobile device such as a tablet

Usability

Unlike the other solutions, FOG isn't available as a Live CD and has to installed. The developers recommend an Ubuntu or CentOS installation, which needs a static IP before running the setup script that automatically fetches the required components. Once it's installed, you can use its browser-based dashboard to image and clone computers across the network.

Malleability

FOG is a complex piece of software and offers plenty of options, with various fields to describe the host images. It can also arrange them into groups for easier management. There are also several options to schedule the imaging process. Besides creating and deploying images, FOG can also be used to debug imaged computers, remotely wipe hosts and more.

Network use

FOG is scaleable and can manage large networks spread over multiple locations, in the same building or around the world. It also supports multicast functionality to deploy multiple machines in one go. On such large installations, you can have multiple FOG installations configured as storage servers, which help take the load off the main imaging server.

Restoration

Deploying an image is a fairly simple process. First ensure that the target computer is registered with the FOG server and has a cloned image associated with it, then all you need to do is to create a deploy image task using the web interface.

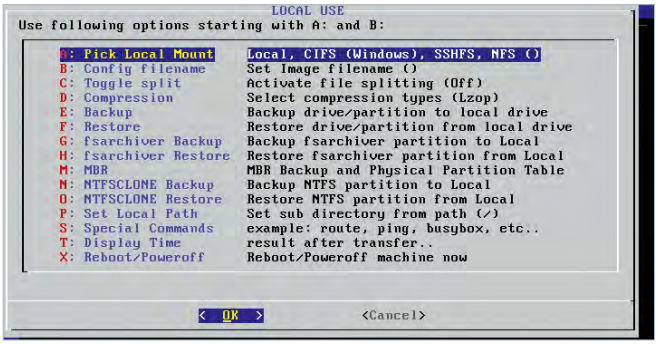
Overall

Apart from the minor inconvenience of installing it before you can use it to clone computers, FOG trumps its competition on all other fronts. Its web front-end makes light work of the entire process.

9

G4L

A comprehensive disk cloner with more dexterity than its competition



G4L includes several useful disk-related utilities, and can directly clone disks without jumping through hoops to create an image

Usability

As with Clonezilla, the interface of the G4L Live CD is text-based and written in ncurses. This one, however, is much more verbose. Creating and restoring images takes you through several wizards, each with quite a few steps. On the plus side, the options are well explained and shouldn't leave you scrabbling about for extra clarification mid-task.

Malleability

G4L's interface has several options to help you fine-tune settings while creating and restoring images. It offers separate menus for imaging individual partitions and complete disks. It enables you to transfer images to remote destinations on the network, and you can also compress images and split them up for easier storage on smaller drives.

Network use

In addition to local images, you can use G4L to create and restore images over the network using UDPcast. Navigate the live environment to set up one machine as a multicast server, which then broadcasts images stored locally or elsewhere on the network. Then use the Live CD on other computers to turn them into 'receivers' for installing the broadcast images.

Restoration

The restoration process is similar to the imaging process. You point the tool to the saved image and the disk/partition where you want to clone the image. Surprisingly, you need to manually select the compression algorithm that you used for creating the clone.

Overall

G4L requires more work than some of the other options here, but offers an unparalleled amount of flexibility for imaging and cloning partitions and disks, both locally and over the network.

8

Redo Backup

An eye-catching cloning tool that does what it promises and little else

We found Redo to be very sensitive to hardware changes, and we were only able to successfully restore images to identical hardware

Usability

Redo Backup & Recovery is one of the simplest apps to use. The Live CD boots straight into a graphical app that offers buttons to either backup or restore an image. Clicking either takes you through a simple wizard. When creating backups, it asks you to select the partitions or disks you want to clone, while during restoration you're prompted for the cloned image.

Malleability

The downside of this simplicity is that unlike the other tools on test here, Redo doesn't give you any tweakable options, such as the ability to select a compression algorithm for squeezing the images – which seems a strange omission. It does however include quite a few useful disk utilities that can help you prepare partitions and disks before cloning them.

Network use

Redo can save and retrieve cloned images from network shares and FTP servers across the local network. If you've configured a network share on your LAN, it'll also detect it automatically and attempt to use it. But since it's primarily designed for home users, Redo doesn't support the ability to clone or deploy images to remote machines over the network.

Restoration

Restoring or deploying clones from images is a walk in the park – you can start the process in a couple of clicks. However, Redo can only restore images to same-sized or larger disks, and it'll error out if you point a larger image to a smaller disk.

Overall

The easiest bare-metal cloning option of the lot, Redo takes away a lot of the control over the cloning process in lieu of convenience. Also, it can only act on local machines.

7

In brief: compare and contrast our verdicts

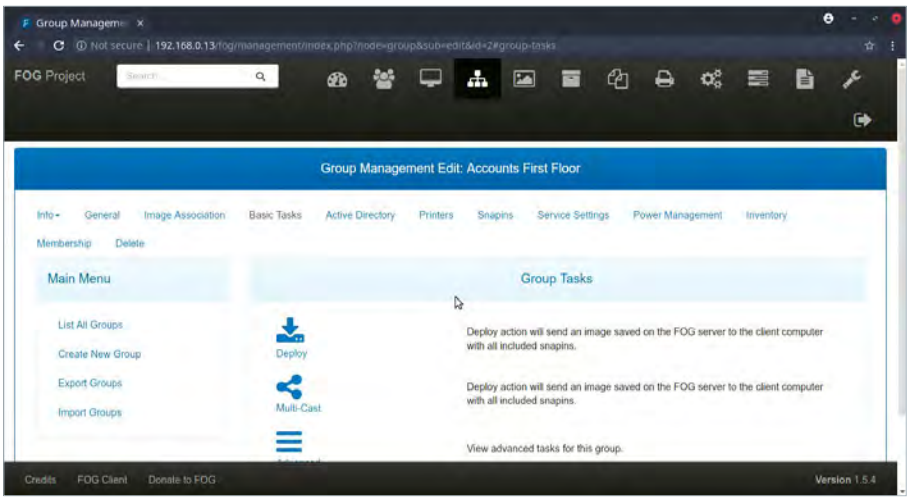
	Clonezilla		FOG Project		G4L		Redo Backup	
Usability	The logically arranged menus with sensible defaults make it a charm to use	9	Not available as a Live CD, but anyone can use its web-based interface with little effort	8	The number of options in its menus can be a little disorientating, especially for new users	7	A straightforward and easy to use app that works even for inexperienced users	9
Malleability	Options for the compression algorithm and encryption of images	8	Offers plenty of useful options to help manage the cloning process with the minimum of fuss	9	There's no dearth of configurable parameters for all the steps of the cloning process	9	It's simple because it doesn't offer the same number of options as others	6
Network use	Works well for larger setups, with two options for cloning over the network	9	Designed from the grounds up for use across setups and networks of all sizes	9	The Live CD helps configure a multicast server to send and receive cloned images	9	Can fetch images from across the network, but can also clone local machines	6
Restoration	The process isn't difficult, but restores only to same-sized or larger disks	7	The process is fairly automated, which is a boon for admins of larger networks	9	Besides the abiity to create images, this tool can also clone partitions and disks on the fly	7	The restoration process is identical to imaging and works as advertised	7
Overall	**An all-round tool that caters to all common cloning and imaging uses cases**	**8**	**It takes a bit more effort to get going, but it's definitely worth the effort**	**9**	**The cumbersome menus give access to a wealth of tweakable parameters**	**8**	**Ideal solution for first timers, but a bit too simple for more experienced ones**	**7**

AND THE WINNER IS...

FOG Project

We can present a use case for each and every one of the cloning tools in this group test. For example, Redo Backup & Recovery is an ideal intuitive solution for individual home-users. But while it did work flawlessly in our tests, the fact that it hasn't been updated for quite a while prevents us from recommending it. At the other end of the spectrum is G4L, which offers a bit too many options in our opinion. Critical administrative software, such as disk cloning tools, require a certain degree of comforting familiarity, and new users will take quite some time to get there with G4L.

In our opinion, Clonezilla and FOG Project are the two viable options that scale well. They can be used by individual users on their home network just as easily as by admins of larger networks. While both do similar tasks, they go about it very differently. The most visible differences between the two solutions are their user interfaces and the fact that Clonezilla runs off a Live CD, while FOG requires setting up a server. FOG is also more comprehensive in that it can perform several tasks other than imaging, such as powering up a computer remotely using Wake On LAN, and installing applications remotely via a feature called snap-ins which helps install software on Windows installations.



The minor effort required to set up FOG quickly redeems itself, especially on large diverse networks

If you're choosing between Clonezilla and FOG and you have machines that aren't connected to a network, you have no option but to use Clonezilla. Clonezilla's network options (SE and Lite) make the most sense for networks with clients that don't require much administration after being imaged, such as those in libraries. But if you need to look after machines once they've been cloned, it's best to go with FOG as its post-imaging options are excellent.

Mayank Sharma

Become an Apple Genius
Learn to troubleshoot Mac problems yourself
WIN!
A portable Mac display worth £225
NEW MacBook Pros reviewed!
Save money on iCloud & iTunes
Share across Mac, iPhone and iPad
18 pages of problems solved
The best office apps rated
Smart kit that keeps you cool
Razer Core X
THE UK'S BEST-SELLING APPLE MAGAZINE!
MACFORMAT.COM

Mac FORMAT

Everything you need to live the Apple life

NEW ISSUE ON SALE NOW!

Available from **www.myfavouritemagazines.co.uk/MACsubs**

Above Supremely comfortable headphones that offer superb sound quality at a high price

HARDWARE

Beyerdynamic Amiron Wireless

Price
£699
Website
https://europe.beyerdynamic.com
Specs
Cable length and connector 1.2m, detachable, 4-pole mini stereo jack (3.5mm)
Frequency range 5Hz-40kHz
Supported profiles HSP, HFP, A2DP, AVRCP, GAVDP
Supported codecs aptX, aptX LL, aptX HD, AAC, SBC
Operating range up to 10m
Battery run time over 30hrs
Charging time 2hrs
Battery capacity 1050mAh
Charging port Bluetooth 4.2, USB-C

The best-sounding wireless headphones you can buy – as you might expect from that price

Wireless headphones have been around for years but have never quite taken off with the audiophile crowd as the limitations of Bluetooth codecs and bandwidth left out a lot of audio detail. Thankfully, however, Bluetooth has advanced significantly in the past several years and now we have codecs such as aptX HD and LDAC offering high-res wireless support that enable headphones to cut the cord without losing fidelity. The result of all this progress is the new Beyerdynamic Amiron Wireless, the best-sounding wireless headphones we've ever heard.

The Amiron Wireless features the iconic Beyer look, which is to say the headphones are huge but extremely comfortable. These headphones are circumaural, which means the ear cups envelop your ears rather than sitting on them. This results in supremely comfortable headphones you can wear for hours, even if they're not the lightest at 380g. Since the headphones are so big and don't fold up, they're not ideal for travelling and are better suited for desk use.

The build quality is excellent and the headphones feel like they will last for many years to come. The headphones are made of a mix of plastic and metal to keep weight down. The headband and earpads are made of Alcantara, a synthetic suede that is extremely comfortable and built to last.

The earpads are replaceable, which is good to see. However, they're extremely difficult to put back on as you have to rotate the pads while trying not to unseat the bead. In terms of noise isolation, the earpads do a decent job of keeping some noise out but the headphones are really meant to be worn around the office and at home.

Above We achieved 25 hours of listening time, which is impressive, and the USB-C port means you can recharge them fast

> These are audiophile cans that aim to give listeners exactly what's on the track, rather than exaggerate

For controls, Beyerdynamic decided to go with a large touchpad on the right earcup. We usually dread using touch controls as they rarely work 100 per cent of the time, but not in this case: the large trackpad prevents mistaken gestures thanks to an increased surface area for you to swipe, and taps registered every time. The right ear-cup also houses the USB-C charging port, power button, 3.5mm headphone jack and mic.

When first listening with the headphones, we weren't blown away by any one aspect of the sound produced. It was only after several weeks of listening and switching between other wireless headphones that we appreciated the levels of detail, wide soundstage and dynamics. No, these aren't headphones that wow you with big bass or sparkling highs – they just sound so clear.

Bass extends low and has decent impact, though it won't satisfy those heathens who like excessive, thudding bass. Then again, these are audiophile cans that aim to give listeners exactly what's on the track, rather than exaggerate any one part. Mids are effortless and do a good job of highlighting vocals and acoustic instruments, such as the guitar and cello, while highs have great detail, extension and energy. The headphones never sound too bright or fatiguing, even after hours of listening. More impressively, the headphones sound exactly the same whether wired or wireless. When wired, you'll want a decent amp to drive them, though they are quite acceptable used with a smartphone at a pinch.

In terms of codec support, the headphones supports AAC, SBC, aptX, aptX HD, and aptX LL. The aptX HD codec supports up to 48kHz/24-bit LPCM audio data, providing higher than CD-quality sound if you have the source material. We loaded up Radiohead's *A Moon Shaped Pool* in 48/24 on a OnePlus 5T (which supports aptX HD) and the headphones sounded excellent, with no stuttering whatsoever. However, you'll definitely want to go wired if you're listening to source material that's higher resolution than 48/24.

Battery life is rated for 30 hours of listening and we achieved around 25 hours at around 50-75 per cent volume – which is well above the competition. Not only do these headphones last a long time, but they charge up relatively quickly thanks to USB-C.

As great as they sound, there are some compromises with these headphones, such as their bulky design and average noise isolation, making them less than ideal for travel. They're also very expensive, but that price isn't too crazy when you consider that the DT 1770 Pro cost £430. If you must have the best sound in a wireless package, the Amiron Wireless headphones are it.

Lewis Leong

Pros

Offers excellent build quality and comfort with a long battery life. The sound is detailed, dynamic and spacious and sounds just as good wireless as when wired.

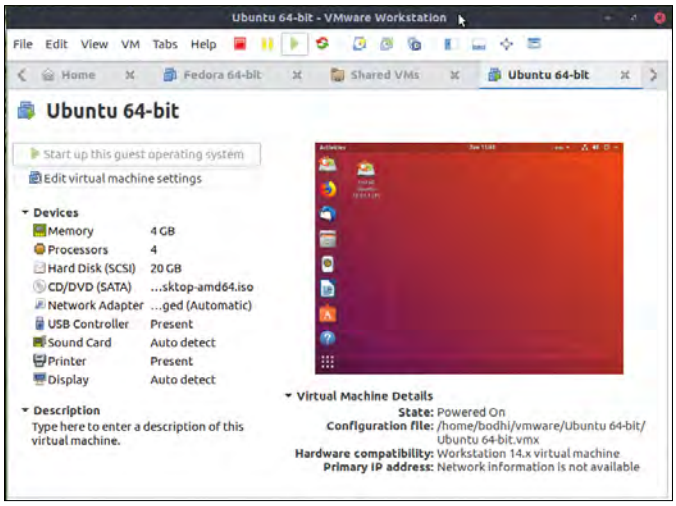
Cons

Not ideal for travel because of their bulky design and merely average exterior noise reduction, while the mobile app needs work. Then there's that price...

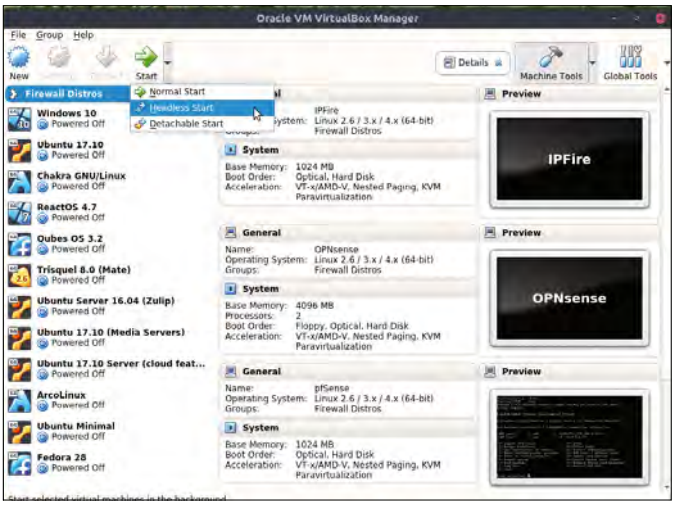
Summary

The best-sounding wireless headphones you can buy, full stop. Sound is spacious, detailed, and makes you want to rediscover your music library. Their bulky design and average noise isolation do not make them a good option for travel, however.

9



Above You can run VMware Workstation Pro as a server so that you can share VMs with other users



Above Both Workstation apps enable you to create VMs with up to 16 CPUs and 64GB of RAM

VIRTUALISATION TOOLS

VMware Workstation 14.1.3 vs VirtualBox 5.2.18

Specs
VMWARE WORKSTATION
Price: £220
CPU: 1.3GHz, 64-bit
RAM: 2GB, 4GB recommended
Storage: 1.2GB
Available from
www.vmware.com

Specs
VIRTUALBOX
CPU: 32-bit or 64-bit with SSE2 support
RAM: 512MB
Storage: 30MB
Available From
www.virtualbox.org

Does the closed source proprietary app offer any real advantage over its open source peer?

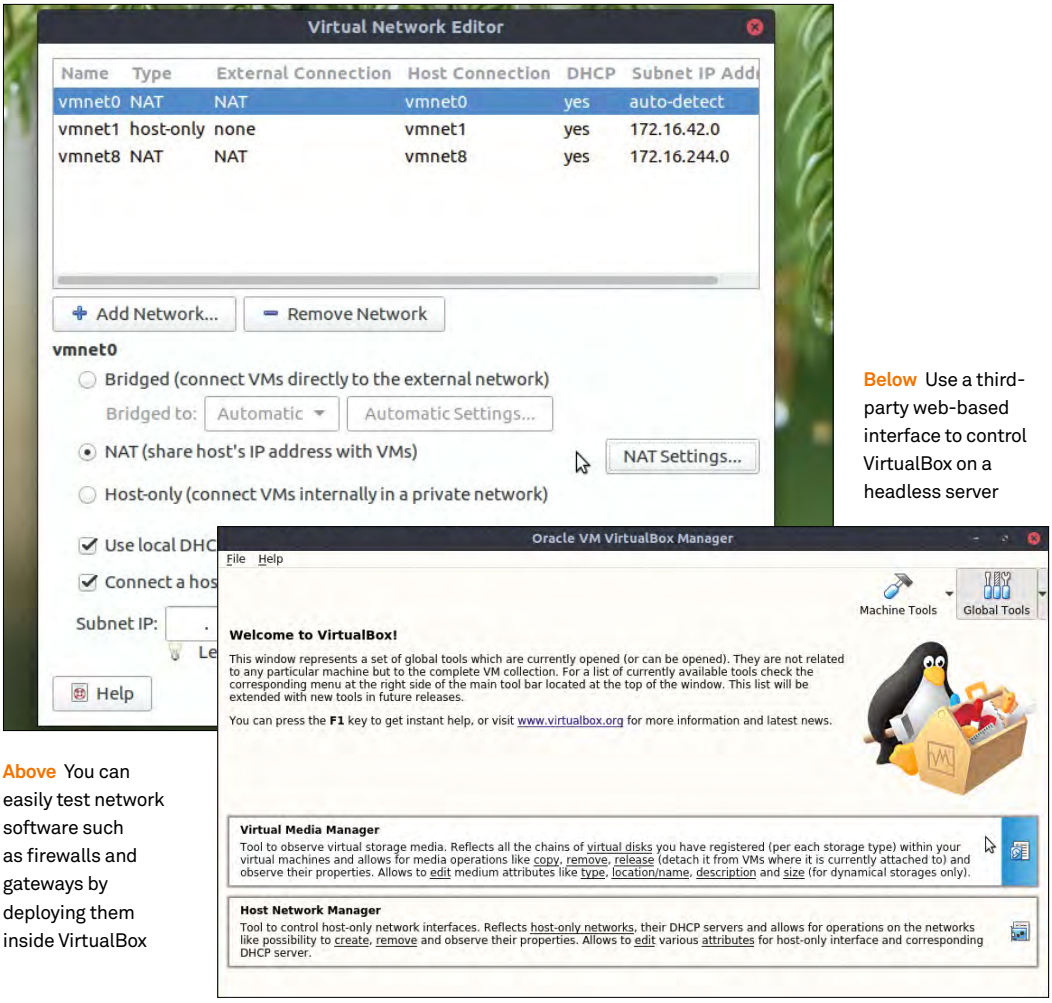
VMware Workstation is the company's flagship desktop virtualisation brand and includes Workstation Player and Workstation Pro. You can use the feature-limited VMware Workstation Player for free for non-commercial use, while the full-featured Workstation Pro is available for a hefty £220. Workstation Player uses the same closed source hypervisor technology platform as Workstation Pro. Despite the price difference both products receive the same performance updates, bug fixes and updated support for new hardware and OS support.

Both Workstation products include all the basic features you'd expect from a desktop virtualisation product. They support over 200 guest OSes, including the Windows 10 Spring Creators Update and Ubuntu 18.04, as both guest and host. Workstation Player enables you to share files between the host and guest and run the VMs in different view modes. Once you've installed the VMware Tools extensions inside a VM, it will support DX10 and OpenGL 3.3. The VMs also support USB 3.0 devices and 4K displays along with a wide range of virtual devices.

On the Linux desktop, the VMware products compete with VirtualBox. Of the two, Workstation Player doesn't offer enough to stand up to the functionality offered by VirtualBox. For instance, you can't create encrypted VMs in Player, while you can in VirtualBox. Similarly, Player doesn't enable you to run multiple VMs at once, which again isn't an issue with VirtualBox.

More importantly, useful features such as snapshots and cloning that are part of VirtualBox are only available in Workstation Pro. Also, while you can control VirtualBox from the VBoxManage command-line interface, the ability to control Workstation with its CLI isn't available to users of the free Player.

Besides their costs, the other major difference between the Workstation bundle and VirtualBox is that the latter is open source, for the most part. Some of VirtualBox's useful functions – such as disk image encryption, support for USB 3.0 devices and bi-directional sharing between the guest and the host – do require the proprietary Extension Pack add-on. While home users can use this for free, commercial VirtualBox deployments need to purchase a licence to use it.


Virtual Network Editor
Name Type External Connection Host Connection DHCP Subnet IP Add
vmnet0 NAT NAT vmnet0 yes auto-detect
vmnet1 host-only none vmnet1 yes 172.16.42.0
vmnet8 NAT NAT vmnet8 yes 172.16.244.0
Add Network...
Remove Network
vmnet0
Bridged (connect VMs directly to the external network)
Bridged to: Automatic
Automatic Settings...
NAT (share host's IP address with VMs)
NAT Settings...
Host-only (connect VMs internally in a private network)
Use local DHC
Connect a hos
Subnet IP:
Help
Oracle VM VirtualBox Manager
File Help
Machine Tools
Global Tools
Welcome to VirtualBox!
This window represents a set of global tools which are currently opened (or can be opened). They are not related to any particular machine but to the complete VM collection. For a list of currently available tools check the corresponding menu at the right side of the main tool bar located at the top of the window. This list will be extended with new tools in future releases.
You can press the F1 key to get instant help, or visit www.virtualbox.org for more information and latest news.
Virtual Media Manager
Tool to observe virtual storage media. Reflects all the chains of virtual disks you have registered (per each storage type) within your virtual machines and allows for media operations like copy, remove, release (detach it from VMs where it is currently attached to) and observe their properties. Allows to edit medium attributes like type, location/name, description and size (for dynamical storages only).
Host Network Manager
Tool to control host-only network interfaces. Reflects host-only networks, their DHCP servers and allows for operations on the networks like possibility to create, remove and observe their properties. Allows to edit various attributes for host-only interface and corresponding DHCP server.

Above You can easily test network software such as firewalls and gateways by deploying them inside VirtualBox

Below Use a third-party web-based interface to control VirtualBox on a headless server

> "The ability to control Workstation with its CLI isn't available to users of the free Player, unlike VirtualBox"

On the enterprise front, Workstation Pro does have an upper hand when it comes to interoperating with other enterprise-specific components, particularly those from VMware. That's mostly down to the fact that while Workstation has been reared for the enterprise desktop, VirtualBox is primarily designed for the everyday desktop user, with features that also make sense in a commercial deployment.

VirtualBox ships with features that make it a viable option for small data centres and an ideal platform to test server deployments, upgrades, and other things before physical rollout. For instance, while VirtualBox VMs are isolated from the network by default, you can run a server inside a VM and set up port forwarding to make sure the server is reachable from outside. Like Workstation Pro, it also includes a network editor tool that enables you to create complex network configurations. VirtualBox also has useful storage emulation features such as NCQ acceleration for raw disks, SSD pass-through and controllers for PATA, SAS and iSCSI disks.

Another interesting difference between the Workstation bundle and VirtualBox is the number of Windows-only features in each. Here too the Workstation bundle outscores VirtualBox, with features such as virtualisation-based security available exclusively for Windows 10 and Windows Server guests. Even some of Workstation's best desktop-centric features, such as Easy Install, are only available to a handful of Linux guests.

That's not to say VirtualBox is without faults. One of its biggest criticisms is the proprietary nature of its extensions add-on. Another prominent limitation is its inability to assign more that 128MB of video memory to guests.

Mayank Sharma

VMware Workstation

Pros

Works with a range of physical and virtual devices on a wide variety of host and guest operating systems.

Cons

The prohibitive cost of Workstation Pro, and the lack of several marquee features in the free Workstation Player.

Virtualbox

Pros

An impressive list of features for both desktop and advanced users, masked behind an intuitive interface.

Cons

Some useful features require the proprietary add-on, and there's limited support for guests outside of Windows and Linux.

Summary

VMware Workstation has the potential to serve admins well in a commercial setting depending on an enterprise's virtualisation requirements and needs. It's a different story on the desktop front where VirtualBox's rich feature-set means it's a much better option than Workstation Player, obliterating the competition.

VMware Workstation 6

VirtualBox 9

BASH HISTORY

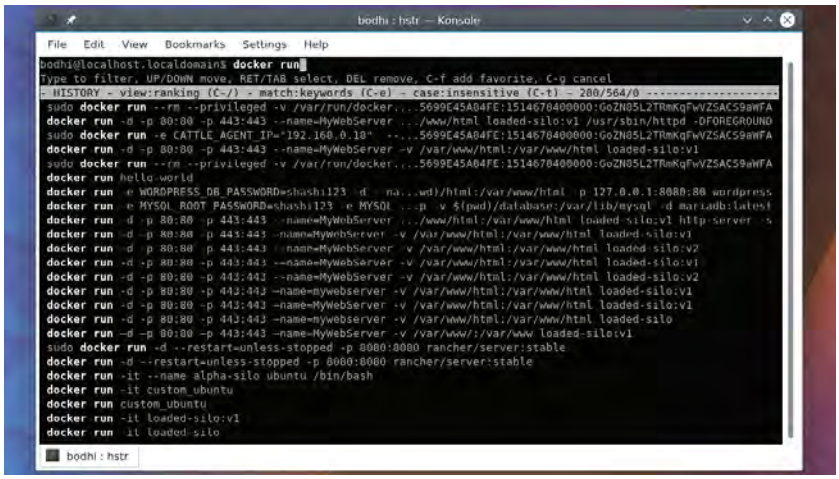
HSTR 1.27.0

Try a useful new visualisation of your Bash history

Ctrl+R is an often-used shortcut by anyone who works with the command line. It brings up a searchable view of the command history, and saves a lot of time re-entering long commands. The HSTR command line utility improves the command completion feature and makes it more efficient than Bash's default. The utility reads through the `.bash_history` file to provide suggestion functionality to filter the commands. Additionally, you can use the **Ctrl+/** key combination to display the most frequently used commands. Press the keys again to cycle through several predefined sorting orders, such as the length and time-stamps of the commands.

Besides the usual completion feature, HSTR enables you to remove commands (such as those with typos or containing sensitive information like passwords) and also allows you to bookmark often used ones for faster retrieval.

HSTR isn't available in the repositories of any distributions, but its developer packages it for many popular distributions including Fedora, Ubuntu, OpenSUSE, Gentoo, Arch and others. Follow the simple instructions on GSTR's GitHub page in order to install the utility and then invoke it using the **hh** or `hstr` command.


bodhi : hstr — Konsole
File Edit View Bookmarks Settings Help
bodhi@localhost.localdomains docker run
Type to filter, UP/DOWN move, RET/TAB select, DEL remove, C-f add favorite, C-g cancel
HISTORY - view:ranking (C-/) - match:keywords (C-e) - case:insensitive (C-t) - 280/564/0
sudo docker run --rm --privileged -v /var/run/docker...5699E45A04FE:1514678400000:GoZN05L2TRmKqFwVZSAC59aWFA
docker run -d -p 80:80 -p 443:443 --name=MyWebServer .../www/html loaded-silo:v1 /usr/sbin/httpd -DFOREGROUND
sudo docker run -e CATTLE_AGENT_IP="192.168.0.10" 5699E45A04FE:1514678400000:GoZN05L2TRmKqFwVZSAC59aWFA
docker run -d -p 80:80 -p 443:443 --name=MyWebServer -v /var/www/html:/var/www/html loaded-silo:v1
sudo docker run --rm --privileged -v /var/run/docker....5699E45A04FE:1514678400000:GoZN05L2TRmKqFwVZSAC59aWFA
docker run hello-world
docker run -e WORDPRESS_DB_PASSWORD=shashi123 -d na...wd)/html:/var/www/html -p 127.0.0.1:8080:80 wordpress
docker run -e MYSQL_ROOT_PASSWORD=shashi123 -e MYSQL ...p -v $(pwd)/database:/var/lib/mysql -d mariadb:latest
docker run -d -p 80:80 -p 443:443 --name=MyWebServer .../www/html:/var/www/html loaded-silo:v1 http-server -s
docker run -d -p 80:80 -p 443:443 --name=MyWebServer -v /var/www/html:/var/www/html loaded-silo:v1
docker run -d -p 80:80 -p 443:443 --name=MyWebServer -v /var/www/html:/var/www/html loaded-silo:v2
docker run -d -p 80:80 -p 443:443 --name=MyWebServer -v /var/www/html:/var/www/html loaded-silo:v1
docker run -d -p 80:80 -p 443:443 --name=MyWebServer -v /var/www/html:/var/www/html loaded-silo:v2
docker run -d -p 80:80 -p 443:443 --name=mywebserver -v /var/www/html:/var/www/html loaded-silo:v1
docker run -d -p 80:80 -p 443:443 --name=MyWebServer -v /var/www/html:/var/www/html loaded-silo:v1
docker run -d -p 80:80 -p 443:443 --name=mywebserver -v /var/www/html:/var/www/html loaded-silo
docker run -d -p 80:80 -p 443:443 --name=MyWebServer -v /var/www/:/var/www loaded-silo:v1
sudo docker run -d --restart=unless-stopped -p 8080:8080 rancher/server:stable
docker run -d --restart=unless-stopped -p 8080:8080 rancher/server:stable
docker run -it --name alpha-silo ubuntu /bin/bash
docker run -it custom_ubuntu
docker run custom_ubuntu
docker run -it loaded-silo:v1
docker run -it loaded-silo
bodhi : hstr

Above While HSTR can be used straight out of the box, the utility offers configurable parameters

Pros

A refreshing new way of parsing through the CLI history that helps to avoid typing repetition.

Cons

None. It doesn't replace **Ctrl+R**, so you can revert to it if you find HSTR cumbersome to use.

Great for...

Quickly accessing frequent commands.

https://github.com/dvorka/hstr

COLLABORATIVE SERVER

Mattermost 5.2

Billed as the open source alternative to Slack

Chances are that if you've asked for recommendation for an open source alternative to Slack which you can host on your own private cloud, you've been pointed to Mattermost. And for good reason – Mattermost has an array of useful features to help enhance communications over a network. It not only offers the usual group chat features such as persistent history and private chat-rooms, but goes a step further with options such as the ability to reply messages and organise them as threads, which is one of its most productive features.

It also supports a variety of built-in and customisable actionable commands, much like IRC. Mattermost enables users to edit and delete messages and create private rooms and also have private one-on-one conversations. Users can add multimedia to messages, which can be viewed inline and downloaded.

Developers will appreciate Mattermost's proper markdown support and various well-documented formatting options. It supports syntax highlighting for over a dozen programming languages, and offers multiple code themes.

The latest version overhauls the plug-ins system, which is still in beta. As with any piece of server software, deploying Mattermost can be involved process, but the procedure is explained in detail on the website. Once it's been deployed, users can interact with the server via desktop apps for Windows, Linux and macOS, as well as mobile apps for iOS and Android.

Pros

Feature-rich server that offers multiple deployment schemes and has cross-platform apps.

Cons

The Linux app is in beta as are some of its functions such as video-calling and plug-ins.

Great for...

Any network which is shared mostly by software developers, although everyone should be able to benefit from it.

www.mattermost.org

BACKUP SERVER

Tardis 1.0.2

A command-line lightweight backup server

Tardis pitches itself as a free alternative to Apple's Time Machine backup app. However these similarities refer to features only rather than the interface. Interactions with Tardis, unlike the swishly graphical Time Machine, take place via the command line. This isn't a criticism, rather just a heads-up to those who might be misled by the description on the website.

Beyond appearances, however, Tardis does exactly what it promises. It takes incremental backups and has simple CLI utilities to check, compare and recover from the backed-up files. The tool runs in a client-server mode, and the server is light enough to be able to run on the older Raspberry Pi 2. You can also choose to run both server and client on the same computer, which is the most likely installation for use on a single machine.

Tardis is written in Python and the installation as described in its GitHub page is straightforward. The server is controlled via a single config file and the installation comes with a template to get you started. The only real setting you have to edit is the location of the backups before you start the server. You'll also have to create a user named tardis.

When you call on the server from the client on a remote machine, you can add a password to encrypt the backups. Everytime you run the tool it'll create a new time-stamped backup set. Recovering the data from the backup set is also fairly simple. Alternatively, you can mount the dataset as a complete file system.

Pros

Does encrypted, incremental remote backups with interesting recovery options.

Cons

Has a slight learning curve and requires a read through the docs to be used effectively.

Great for...

Scheduling incremental backups on several machines (or even just on one).

https://github.com/koldinger/Tardis

AUDIO PLAYER

Audacious 3.10

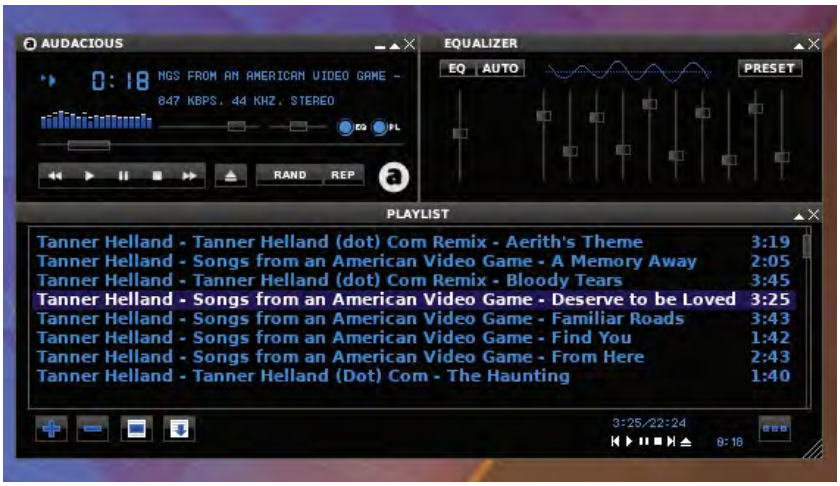
A good-looking, lightweight, feature-rich music player

The Linux desktop has no dearth of media players. What sets Audacious apart from others is its lightweight design and its set of useful plug-ins. This BSD-licensed media player supports a wide variety of audio formats including OGG, MP3, FLAC, AAC and more. Besides playback, you can use the app to create playlists and edit metadata.

Audacious is available in the official repositories of most desktop distributions such as Fedora and Ubuntu. The app comes with two UIs: a GTK+ interface and a classic Winamp clone. The latter offers several different skins. By default, however, the player uses the multi-tabbed GTK interface.

One of the strong points of Audacious is its plug-ins, many of which have several settings of their own. The audacious-plugins package in Fedora doesn't include all the packages due to Fedora's packaging policy, so you'll have to fetch them from the RPM Fusion repositories.

In addition to influencing playback, the plug-ins affect several other aspects of the player such as Visualisation, Playlists and Output. The latest release implements some useful new features such as the ability to remember shuffle history between app restarts to avoid repeating songs.


AUDACIOUS
0:18
NGS FROM AN AMERICAN VIDEO GAME -
847 KBPS, 44 KHZ, STEREO
RAND
REP
EQUALIZER
EQ
AUTO
PRESET
PLAYLIST
Tanner Helland - Tanner Helland (dot) Com Remix - Aerith's Theme 3:19
Tanner Helland - Songs from an American Video Game - A Memory Away 2:05
Tanner Helland - Tanner Helland (dot) Com Remix - Bloody Tears 3:45
Tanner Helland - Songs from an American Video Game - Deserve to be Loved 3:25
Tanner Helland - Songs from an American Video Game - Familiar Roads 3:43
Tanner Helland - Songs from an American Video Game - Find You 1:42
Tanner Helland - Songs from an American Video Game - From Here 2:43
Tanner Helland - Tanner Helland (Dot) Com - The Haunting 1:40
3:25/22:24
0:18

Above One of the useful features of the player is the built-in support for gapless playback.

Pros

Lightweight, cross-platform, feature-rich audio player that can be extended with plug-ins.

Cons

Does what it's meant to, but still isn't as versatile as some of the other desktop apps out there.

Great for...

Anyone looking for a modern audio player.

https://audacious-media-player.org

Get your listing in our directory
To advertise here, contact Chris
chris.mitchell@futurenet.com | +44 01225 68 7832 (ext. 7832)

RECOMMENDED

Hosting listings

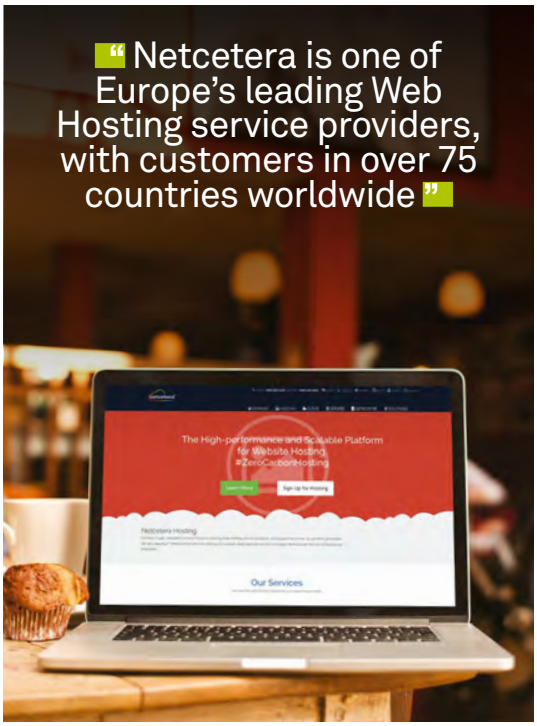
Featured host:
www.netcetera.co.uk
03330 439780

About us

Formed in 1996, Netcetera is one of Europe's leading web hosting service providers, with customers in over 75 countries worldwide. It is a leading IT infrastructure provider offering co-location, dedicated servers and managed infrastructure services to businesses worldwide.

What we offer

- Managed Hosting
  A full range of solutions for a cost-effective, reliable, secure host
- Dedicated Servers
  Single server through to a full racks with FREE setup and a generous bandwidth allowance
- Cloud Hosting
  Linux, Windows, hybrid and private cloud solutions with support and scaleability features
- Datacentre co-location from quad-core up to smart servers, with quick setup and full customisation



Five tips from the pros

01 **Optimise your website images**
When uploading your website to the internet, make sure all of your images are optimised for the web. Try using **jpegmini.com** software; or if using WordPress, install the EWWW Image Optimizer plugin.

02 **Host your website in the UK**
Make sure your website is hosted in the UK, and not just for legal reasons. If your server is located overseas, you may be missing out on search engine rankings on google.co.uk – you can check where your site is based on **www.check-host.net**.

03 **Do you make regular backups?**
How would it affect your business if you lost your website today? It's vital to always make your own backups; even if your host offers you a backup solution, it's important to take responsibility for your own data and protect it.

04 **Trying to rank on Google?**
Google made some changes in 2015. If you're struggling to rank on Google, make sure that your website is mobile-responsive. Plus, Google now prefers secure (HTTPS) websites. Contact your host to set up and force HTTPS on your website.

05 **Avoid cheap hosting**
We're sure you've seen those TV adverts for domain and hosting for £1! Think about the logic... for £1, how many clients will be jam-packed onto that server? Surely they would use cheap £20 drives rather than £1k+ enterprise SSDs? Remember: you do get what you pay for.

Testimonials

David Brewer
"I bought an SSL certificate. Purchasing is painless, and only takes a few minutes. My difficulty is installing the certificate, which is something I can never do. However, I simply raise a trouble ticket and the support team are quickly on the case. Within ten minutes I hear from the certificate signing authority, and approve. The support team then installed the certificate for me."

Tracy Hops
"We have several servers from Netcetera and the network connectivity is top-notch – great uptime and speed is never an issue. Tech support is knowledge and quick in replying – which is a bonus. We would highly recommend Netcetera."

J Edwards
"After trying out lots of other hosting companies, you seem to have the best customer service by a long way, and all the features I need. Shared hosting is very fast, and the control panel is comprehensive..."

Supreme hosting

CWCS MANAGED HOSTING — SUPREME HOSTING. SUPREME SUPPORT.

www.cwcs.co.uk
0800 1 777 000

CWCS Managed Hosting is the UK's leading hosting specialist. It offers a fully comprehensive range of hosting products, services and support. Its highly trained staff are not only hosting experts, it's also committed to delivering a great customer experience and is passionate about what it does.

- Colocation hosting
- VPS
- 100% Network uptime

Value hosting

elastichosts

elastichosts.co.uk
02071 838250

ElasticHosts offers simple, flexible and cost-effective cloud services with high performance, availability and scalability for businesses worldwide. Its team of engineers provide excellent support around the clock over the phone, email and ticketing system.

- Cloud servers on any OS
- Linux OS containers
- World-class 24/7 support

Small business host

HOSTPAPA

www.hostpapa.co.uk
0800 051 7126

HostPapa is an award-winning web hosting service and a leader in green hosting. It offers one of the most fully featured hosting packages on the market, along with 24/7 customer support, learning resources and outstanding reliability.

- Website builder
- Budget prices
- Unlimited databases

Enterprise hosting:

2020Media www.2020media.com

www.2020media.com | 0800 035 6364

WordPress comes pre-installed for new users or with free managed migration. The managed WordPress service is completely free for the first year.

We are known for our "Knowledgeable and excellent service" and we serve agencies, designers, developers and small businesses across the UK.

Budget hosting:

HETZNER ONLINE

www.hetzner.de/us | +49 (0)9831 5050

Hetzner Online is a professional web hosting provider and experienced data-centre operator. Since 1997 the company has provided private and business clients with high-performance hosting products, as well as the necessary infrastructure for the efficient operation of websites. A combination of stable technology, attractive pricing and flexible support and services has enabled Hetzner Online to continuously strengthen its market position both nationally and internationally.

- Dedicated and shared hosting
- Colocation racks
- Internet domains and SSL certificates
- Storage boxes

SSD web hosting

bargainhost

www.bargainhost.co.uk
0843 289 2681

Since 2001, Bargain Host has campaigned to offer the lowest-priced possible hosting in the UK. It has achieved this goal successfully and built up a large client database which includes many repeat customers. It has also won several awards for providing an outstanding hosting service.

- Shared hosting
- Cloud servers
- Domain names

Value Linux hosting

PATCHMAN WEB HOSTING

patchman-hosting.co.uk
01642 424 237

Linux hosting is a great solution for home users, business users and web designers looking for cost-effective and powerful hosting. Whether you are building a single-page portfolio, or you are running a database-driven ecommerce website, there is a Linux hosting solution for you.

- Student hosting deals
- Site designer
- Domain names

Fast, reliable hosting

:BYTEMARK

www.bytemark.co.uk
01904 890 890

Founded in 2002, Bytemark are "the UK experts in cloud & dedicated hosting". Its manifesto includes in-house expertise, transparent pricing, free software support, keeping promises made by support staff and top-quality hosting hardware at fair prices.

- Managed hosting
- UK cloud hosting
- Linux hosting

Free Resources

Welcome to Filesilo!

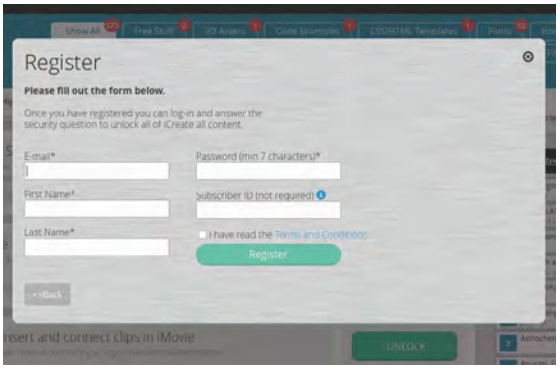
Download the best distros, essential FOSS and all our tutorial project files from your FileSilo account

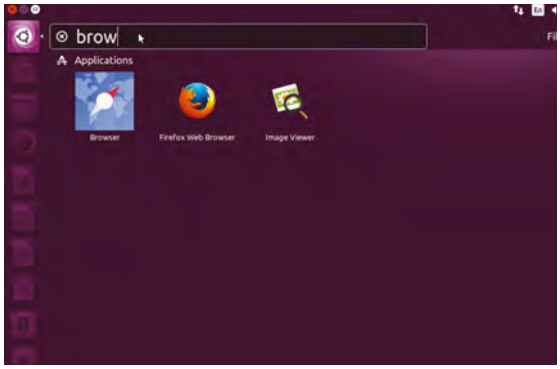
WHAT IS IT?

Every time you see this symbol in the magazine, there is free online content that's waiting to be unlocked on FileSilo.

WHY REGISTER?

- Secure and safe online access, from anywhere
- Free access for every reader, print and digital
- Download only the files you want, when you want
- All your gifts, from all your issues, all in one place





1. UNLOCK YOUR CONTENT

Go to **www.filesilo.co.uk/linuxuser** and follow the instructions on screen to create an account with our secure FileSilo system. When your issue arrives or you download your digital edition, log into your account and unlock individual issues by answering a simple question based on the pages of the magazine for instant access to the extras. Simple!

2. ENJOY THE RESOURCES

You can access FileSilo on any computer, tablet or smartphone device using any popular browser. However, we recommend that you use a computer to download content, as you may not be able to download files to other devices. If you have any problems with accessing content on FileSilo, take a look at the FAQs online or email our team at **filesilohelp@futurenet.com**.

Free for digital readers too!
Read on your tablet, download on your computer

Log in to **www.filesilo.co.uk/linuxuser**

Packed with distros and resources

Lost the disc or have the digital edition? Grab all the distros, software and tutorial code for this issue from the FileSilo

This month find...

DISTROS

We've four Linux distros in the FileSilo for you to try. Our first is Zorin OS 12.4 Core, which is designed as a direct Windows replacement. The remaining three distros complement our main feature on getting the right, light and fast Linux distro for you. We've included the lightweight and minimalist Arch-based distro, ArchLabs 2018.07. Alongside this we've supplied Linux Lite 4.0, a lightweight Windows-like distro. Last, we have Porteus MATE, the perfect Live distro to keep in your pocket.

SOFTWARE & CODE

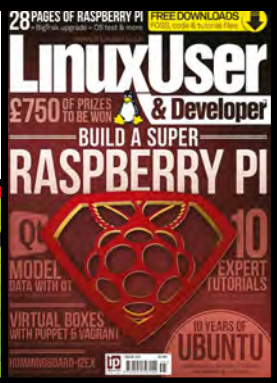
Along with tutorial code, we've uploaded an IDE bundle including the popular web dev editor Brackets 1.13, NetBeans IDE 8.2 and Microsoft's Visual Studio Code 1.26.

FOLLOW US
Facebook: facebook.com/LinuxUserUK
Twitter: @linuxusermag

Left A much younger Jon Masters unleashes his Kernel Column on an unsuspecting world. He's actually written for LU&D for 18 years. Thanks Jon, it's been a pleasure!

FREE DVD! LIGHT DISTROS BUNDLE!
Linux User & Developer
OPTIMISE LINUX
The Future of Moore's Law
Pick the perfect IDE
45 PAGES OF TUTORIALS

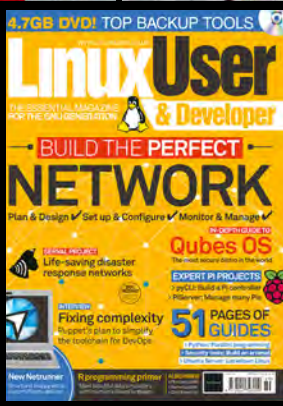
Below Linux User and Developer was relaunched in 2009, and who better to interview but Canonical's Mark Shuttleworth.



So long and thanks for being part of the LU&D community

Go create, code and make. Leave your mark and make the world a better place!

Left We didn't quite make it to 200 issues, but to celebrate our 100th issue, we handed the reins to Jim Zemlin, the executive director of the Linux Foundation... and went to the pub.



THE ESSENTIAL MAGAZINE FOR THE GNU GENERATION

techradar.

The source for tech buying advice

techradar.com

The Electronic Frontier Foundation is the leading nonprofit organization defending civil liberties in the digital world. Founded in 1990, EFF champions user privacy, free expression, and innovation through impact litigation, policy analysis, grassroots activism, and technology development. We work to ensure that rights and freedoms are enhanced and protected as our use of technology grows.

EFF.ORG

ELECTRONIC FRONTIER FOUNDATION

Protecting Rights and Promoting Freedom on the Electronic Frontier

9000